DEVDAGGER: DATA EFFICIENT VISUAL IMITATION LEARNING

KUN HUANG [HUANGKUN@SEAS.UPENN.EDU]
ZHIHAO RUAN [RUANZH@SEAS.UPENN.EDU]

ABSTRACT. DAgger is a well-known online imitation learning algorithm in robot planning and control problems. Traditional DAggers suffer from the problem of high expert querying rate and could be costly under situations where expert querying is expensive. Moreover, traditional DAgger has not yet developed a state-of-the-art solution for vision-based planning and control. We propose DevDAgger, a vision-based data-efficient DAgger algorithm based on CNN and Gaussian Processes (GP) targeting low expert querying rate and visual imitation learning. We demonstrated our algorithm on the simple cart-pole problem and showed that state-based DevDAgger outperforms Vanilla DAgger in both performance and expert querying rate.

1. INTRODUCTION

Imitation learning is commonly used in planning and control in high-dimensional state space with non-linear dynamical model in robotics. It offers robots the ability to manipulate in complex environment without explicitly modelling the environment by imitating the control actions from an expert (i.e., human). Dataset Aggregation (DAgger) [1], in particular, is a popular online imitation learning algorithm that trains the controller while aggregating new data queried from the expert simultaneously. Traditional DAgger uses a neural network to train the controller, requiring lots of expert data which is often costly to obtain in many situations. This can actually be improved by replacing the neural network with Gaussian Process (GP), which is a commonly used model-free data efficient technique for regression tasks. Recent literature also shows a particular interest in vision-based robotic planning and control problems (visuomotor control) [2, 3, 4]. As a consequence, we propose a *Data-Efficient Vision-based DAgger (DevDAgger)* algorithm for imitation learning tasks, combining the advantage of Gaussian Process and latent representation learning with Convolutional Neural Networks (CNNs).

2. RELATED WORK

Imitation learning, as a supervised learning algorithm for robotic planning and control, has long been a significant part in robotic research. A great variety of approaches has been proposed for the traditional imitation learning tasks. On one hand, many approaches has attempted to incorporate *parametric models* to the imitation learning problem, such as dynamic movement primitives (DMP) [5], Gaussian mixture models (GMM) [6], and probabilistic movement primitives (ProMP) [7]. On the other hand, recent literature has also shown an extensive attention to *non-parametric models*,

such as kernelized movement primitives (KMP) [8] and linearly constrained kernelized movement primitives (LC-KMP) [9].

DAgger, as an online imitation learning algorithm, was proposed in 2011 by constantly aggregating data from the expert and imitating expert's behaviors [1]. Recent works have addressed some drawbacks of DAgger by improving the expert data querying efficiency [10]. Uncertainty estimation was also introduced to DAgger by EnsembleDAgger [11] to improve the training-time safety.

Gaussian Processes (GP) itself has been a simple yet powerful non-parametric model for general regression tasks. Traditional exact GP suffers from the limitation of dataset in order to achieve reasonable amount of training time and thus cannot be put into direct use in large scale learning tasks. However, the computational efficiency of exact GP has recently been greatly improved by GPyTorch [12, 13]. By leveraging the power of GPyTorch, exact GP has now become a new option for *non-parametric* imitation learning. EnsembleDAgger also attempts to incorporate ensembles and Monte-Carlo methods as an approximation to exact GP [11].

To summarize, we decided to utilize the power of exact GP in our DAgger solution. We also incorporated the use of CNN to achieve vision-based planning and control as an extra layer to the traditional DAgger pipeline. Meanwhile, although GP suffers from the curse of dimensionality, recent works have shown that mapping the structured high-dimensional inputs to latent space [14] dramatically expands the capability of GP.

3. Approach

Our proposed DevDAgger aims to provide a fast and data-efficient framework for vision-based online imitation learning, or DAgger. As discussed above, the existing variants of DAgger still attempts to use various approximations to GP for uncertainty estimation, and has not yet incorporated vision-based planning and control. Therefore, our proposed DevDAgger is composed of two parts: a novel GP-based probabilistic visuomotor controller and an uncertainty-based decision rule for DAgger to optimize expert query frequency.

3.1. Problem Formulation.

In the context of general imitation learning, one would need the following definitions: a *novice* policy, an *expert* policy, a well-defined state space and a well-defined action space. Imitation learning generally trains the novice to produce action a_t exactly as the expert given some current observation of the state o_t.

DAgger works as an online algorithm for imitation learning. Figure 1 illustrates the workflow of a general DAgger. DAgger collects dataset and trains the novice as the novice acts out in the environment. Given some observations o_t from the environment, both novice and expert would generate their own actions a_{nov} and a_{exp}. A pre-defined decision rule would then choose an action a_t and append the observation-action pair (o_t, a_t) to the dataset. The novice then gets trained and observes a new o_t in the next iteration.

As more and more iteration takes place, the novice gets trained for more and more times and would act more and more similar to the expert. The decision rule decides whether the novice action a_{nov} or the expert action a_{exp} should be added to the dataset for the next training. There should

always be some novice actions in the dataset to ensure that the novice policy is generalizable to unvisited states.

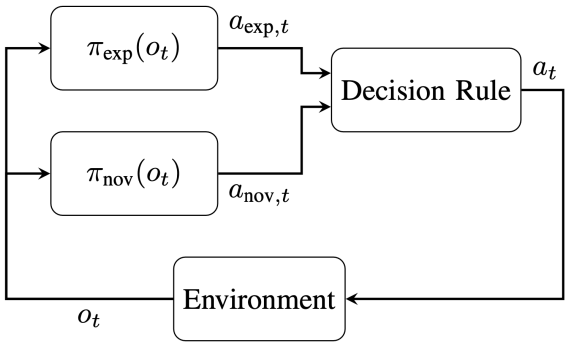

FIGURE 1. Illustration of workflow of general DAgger [11].

3.2. **Imitation Learning with Gaussian Processes.** Our algorithm starts with an observation-action pair (o_t, a_t) from the dataset. We first map the visual inputs o_t into some latent space using the CNN defined in Figure 2. Given the colored visual inputs, we resize the images to be $3 \times 64 \times 64$ and stack the visual inputs from 2 consecutive frames together in the color dimension in order to provide speed information to the CNN, forming a final input of size $6 \times 64 \times 64$. The images are then forwarded through the CNN and produces the corresponding latent space feature points z_t. All convolutional layers use the SAME padding. Although Figure 2 only shows 2 sets of convolution layers, there can be multiple sets of convolution layers in the CNN architecture and this is a hyperparameter that should be tuned according to different environments and robot dynamics models.

The output dimension of the linear layers of the CNN are shrunk by a factor of 4 in each layer. The linear layers would finally produce a latent space feature point z_t according to each o_t.

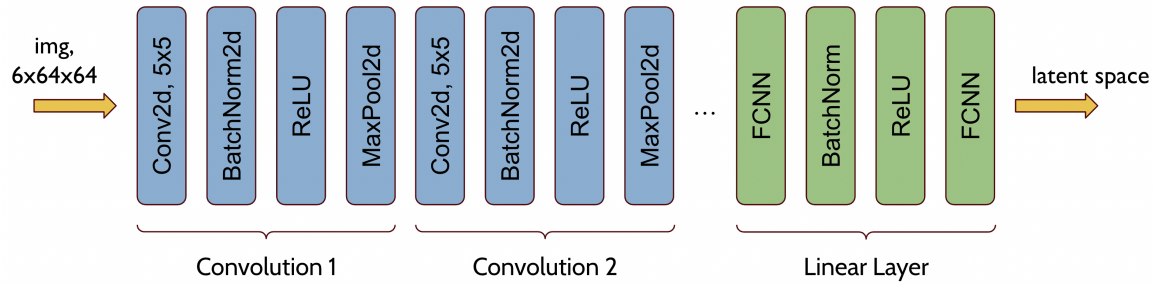

FIGURE 2. Diagram for mapping from image space to GP latent space.

Given the latent space feature points z_t, we then perform a GP regression task over (z_t, a_t) using GPyTorch [13]. The predicted probability distribution of action over latent space from the GP model is then passed into a Gaussian likelihood function, and the predicted novice action a_{nov} is defined as the mean of the likelihood.

3.3. **Uncertainty Estimation.** Similar to EnsembleDAgger [11], we use the uncertainty measurements from GP as a criterion for choosing the novice action over the expert action in the decision rule. The uncertainty measurements from GP is defined as the standard deviation of the likelihood.

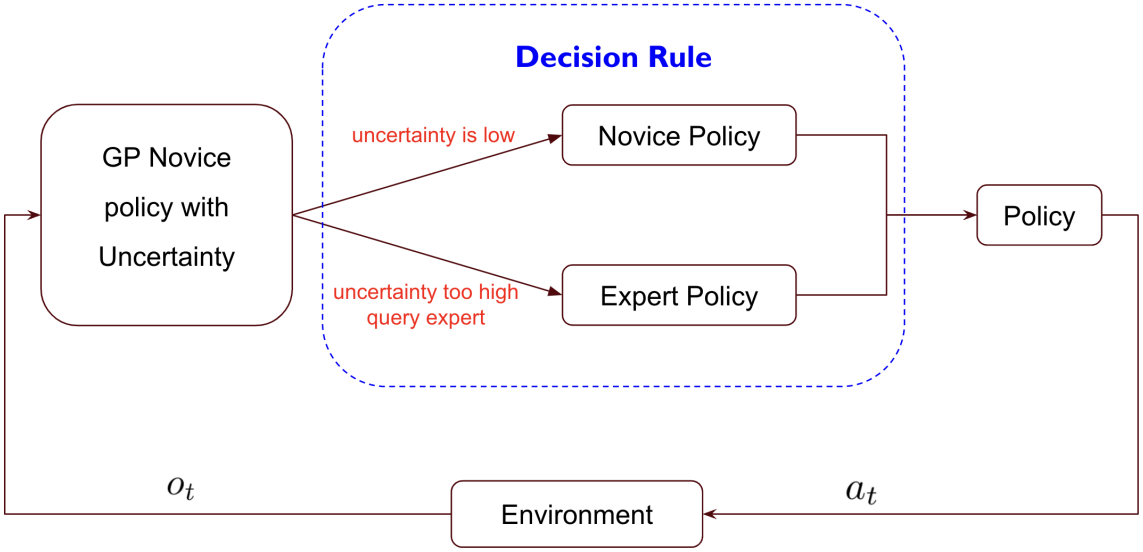

FIGURE 3. Illustration of workflow for DevDAgger.

As shown in Figure 3, the decision rule for DevDAgger is that expert action a_{exp} are only executed and queried for the given o_t whenever the novice is highly uncertain of its own action a_{nov}. Specifically, we set an expert querying threshold for the GP output uncertainty measurement that

$$\text{Uncertainty upper bound} = \alpha \times \sqrt{\text{covar\_module.outputscale} + \text{likelihood.noise}} \tag{1}$$

where α is a tunable hyperparameter. Whenever a_{nov} comes with an uncertainty higher than this upper bound, DevDAgger would query the expert and aggregate a_{exp} to the dataset.

Since GP is accurate in uncertainty measurements, the use of GP in regressing latent space $\mathcal{Z}$ to action space $\mathcal{A}$ provides an accurate estimation of novice's confidence of its own action and thus saves a lot of expert queries than other variants of DAgger [11].

4. EXPERIMENTAL RESULTS

Our experiments focus on imitation learning with Gaussian Processes, the potential of using Deep Kernel Learning (CNN + GP) to estimate uncertainties from images, and eventually DevDAgger. In this section, we present experimental validation for the following claims:

(1) Assuming access to the ground truth states of agents
   (a) Gaussian Processes perform well in terms of action regression and uncertainty estimation.
   (b) The GP-novice policy's output variance is a good measure of dissimilarity between the query state and states in the training dataset.

(c) By purely relying on the GP's output uncertainties in the DevDAgger framework, the novice policy is able to achieve better performance due to high exploration coverage, despite making fewer queries to the expert policy.

(2) When only visual observations are given
   (a) The GP-novice policy is able to learn with significantly more demonstrations.
   (b) The uncertainty estimation is highly inaccurate since the learned latent space is not well structured.
   (c) Due to the above reason, DevDAgger is not able to improve query efficiency when only visual observations are given.

To justify the above claims, we test our proposed approaches on a simple cart-pole system as illustrated in Fig 4. The objective of this task is to balance the inverted pendulum by moving the cart horizontally. The ground truth state of this cart-pole system consists of 4 dimensions $x, \dot{x}, \theta, \dot{\theta}$, where x is the horizontal position of the cart, θ is the angle of the inverted pendulum, and $\dot{x}$ and $\dot{\theta}$ represent the positional velocity and angular velocity respectively. To learn from demonstrations, we make use of a PID controller as the expert policy. Note that this PID controller only considers θ and $\dot{\theta}$ and is invariant to x and $\dot{x}$.

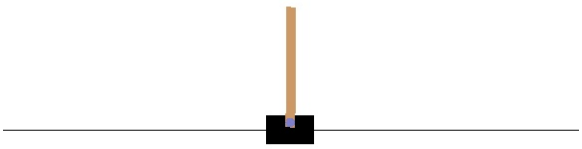

FIGURE 4. An example of the visual observation of the cart-pole system.

4.1. **Imitation Learning & Uncertainty Estimation with Gaussian Processes.**

4.1.1. *State-based Imitation Learning.* When the ground truth states of the cart-pole system are accessible, we use a GP to directly regress the function mapping from states to actions based on the demonstration dataset collected by the expert controller. One of the key benefits of using GP is the learning efficiency, and this is supported by the quantitative results shown in Table 1. The state-based GP controller is able to learn from very few examples (Fig 5:Left), and achieves high accuracies in the state space covered by the training dataset, in the meantime generalizes reasonably well to unseen states (Fig 5:Right).

Furthermore, GP is able to model the epistemic uncertainty (Fig 5:Middle), in which unseen states generally correspond to high uncertainties.

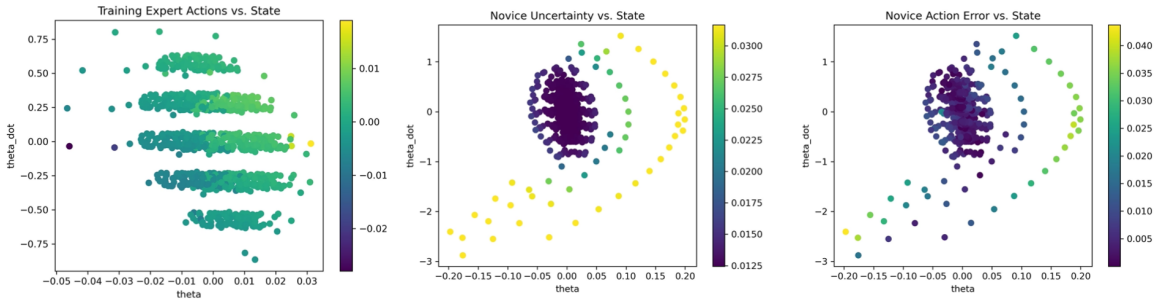

FIGURE 5. State-based imitation learning with GP. Left: visualization of the training demonstration dataset distribution, color represents the expert policy control output (target); Middle: uncertainty estimation from GP on a testing dataset, color represents uncertainty; Right: novice policy control error, color represents the absolute difference between the expert policy action and the novice policy action for a given state.

4.1.2. *Visual Imitation Learning.* To explore the possibilities of using non-parametric models like GP for high dimensional inputs, we experiment Deep Kernel Learning with a simple imitation learning task. Note that doing visuomotor control is an inherently difficult task in the sense that the model must learn to extract visual features. We stack two adjacent images together to feed in velocity information. Compared with state-based IL, visual IL achieves worse performance despite requiring more training data (Table 1).

We further visualize the uncertainty estimation on the testing dataset in Fig 6:Middle, and discover that the epistemic uncertainties are not correctly modeled by Deep Kernel Learning, as two observations correspond to very different uncertainties even though their ground truth states are extremely close.

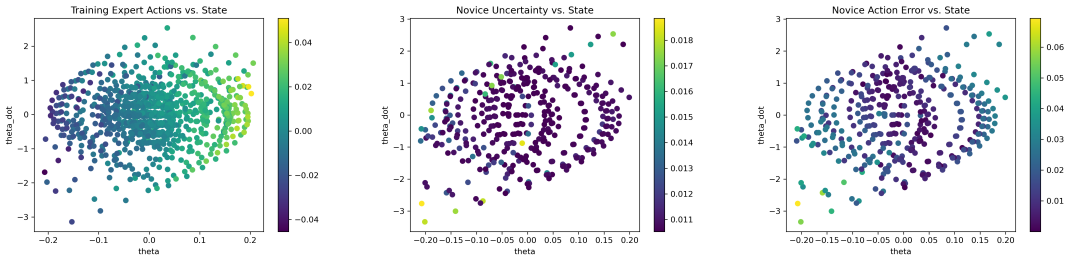

FIGURE 6. Visual imitation learning with GP.

To investigate this issue of uncertainty estimation, following the visualization protocol presented in prior work "Embed to control" [15], we set the CNN output latent space to be 2-dimensional, and visualize the mapping between the state space and the latent space as shown in Fig 7. Ideally the CNN would be forced to extract the true state information from visual observations. However, the Deep Kernel Learning model fails to discover the underlying structure of the state space. Our

hypothesis for this phenomenon is that the image distance does not reflect the actual state distance, thus, without constraints, the model naturally has no clue to learn a structured latent space.

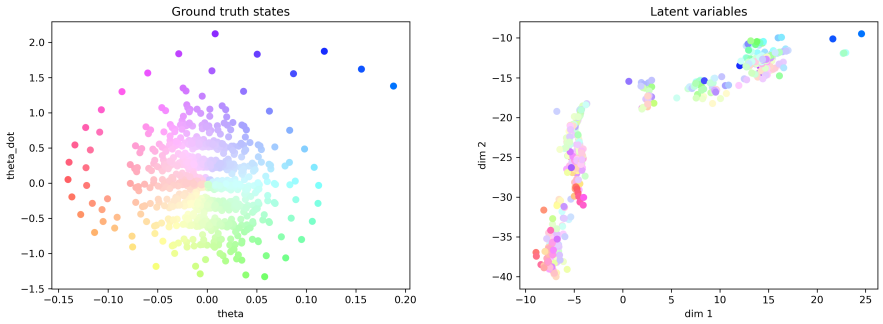

FIGURE 7. Visualization of the latent space learned by the CNN. Left: ground truth states; Right: corresponding learned latent representations. Points in the true state space and the latent space are associated by color.

	Avg Balanced Duration ↑	Expert Query Count ↓
State-based IL	67.1	200.0
State-based DevDAgger	**74.08**	**178.4**
Visual IL	**45.87**	1000.0
Visual Vanilla DAgger ($\gamma = 0.995$)	35.6	1000.0
Visual Vanilla DAgger ($\gamma = 0.999$)	**45.3**	1000.0
Visual DevDAgger	**43.15**	**613.3**

TABLE 1. Quantitative comparison between different approaches in terms of both the average number of time steps during which the inverted pendulum is balanced, and the number of queries made for the expert policy. Each approach is at least tested for 50 trials.

4.2. **DevDAgger: Uncertainty-based Decision Rule for DAgger.** We further experiment with incorporating uncertainty estimation into online imitation learning algorithm DAgger.

4.2.1. *State-base DevDAgger.* The uncertainty-based decision rule is set as followings: if the predicted uncertainty is larger than $\alpha\sigma_{prior}$ where $\alpha = 0.7$, query the expert policy for what action a to take, and aggregate this new state-action pair (s, a) to the current dataset; otherwise, rollout the current novice policy.

With each newly aggregated state-action pair, ideally we would train the GP model again, however, an exact GP model is trained every 30 data collection steps with the aim to improve runtime efficiency.

The quantitative comparison between the state-based IL and the state-based DevDAgger in Table 1 shows that the proposed uncertainty-aware decision rule is able to learn a better policy with fewer expert queries in an online fashion given the ground truth states.

4.2.2. *Visual DevDAgger.* We use the same uncertainty threshold α, and experimented with different latent space dimensions $2, 4, 8, 16$. As discussed in section 4.1.2, the uncertainty estimation of the Deep Kernel Learning model is very problematic due to the unstructured learned latent space. This phenomenon causes the unreasonable behavior of the uncertainty-aware decision rule and fails to improve the performance compared with visual imitation learning (Table 1).

5. Future Work

To resolve the issue with the uncertainty estimation of visual GP imitation learning, a structured latent space has to be enforced by adding constraints to the model. A very simple solution would be to train a regression model that predicts the ground truth states from images. We can also use contrastive learning to enforce the structure in the learned latent space. However, both of these two methods rely on supervision, i.e. providing ground truth states as labels for each image.

Without ground truth states as supervision, proposed by [15], we could train a dynamics model in the latent space and force the dynamical model to be linear. As an extension to [15], a more recent work [16] achieves better results in terms of the latent space structure.

6. Conclusion

We propose DevDAgger, a data-efficient vision-based algorithm for online imitation learning. Compared with other variants of DAgger, DevDAgger takes the advantage of Gaussian Processes (GP) and GPyTorch [13] in robot state space — action space regression tasks for accurate uncertainty measurements. With accurate uncertainty measurements, expert queries can be reduced, which makes the DAgger process data-efficient. CNN is also put in use in order to achieve vision-based robot planning and control, which is of high significance in the ongoing imitation learning research areas.

In our experiments, we tested our DevDAgger algorithm on a simple cart-pole problem and demonstrated that in state-based imitation learning, GP is able to model the epistemic uncertainty well. State-based DevDAgger is also able to outperform Vanilla DAgger both in performance and in expert queries. However, more investigations are needed in visual imitation learning and visual DAgger.

Future work includes enforcing a structured latent space by adding constraints to the model, and training a dynamics model in the latent space to provide more robot-specific information to the DAgger problem.

REFERENCES

[1] S. Ross, G. Gordon, and D. 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*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[2] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Proceedings of The 2nd Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Billard, A. Dragan, J. Peters, and J. Morimoto, Eds., vol. 87. PMLR, 29–31 Oct 2018, pp. 651–673. [Online]. Available: https://proceedings.mlr.press/v87/kalashnikov18a.html

[3] F. Ebert, C. Finn, S. Dasari, A. Xie, A. Lee, and S. Levine, "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control," 2018.

[4] A. Xie, A. Singh, S. Levine, and C. Finn, "Few-shot goal inference for visuomotor learning and planning," in *Conference on Robot Learning*. PMLR, 2018, pp. 40–52.

[5] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Comput.*, vol. 25, no. 2, p. 328–373, feb 2013. [Online]. Available: https://doi.org/10.1162/NECO_a_00393

[6] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intelligent Service Robotics*, vol. 9, no. 1, pp. 1–29, 2016. [Online]. Available: https://doi.org/10.1007/s11370-015-0187-9

[7] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," in *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013. [Online]. Available: https://proceedings.neurips.cc/paper/2013/file/e53a0a2978c28872a4505bdb51db06dc-Paper.pdf

[8] Y. Huang, L. Rozo, J. Silvério, and D. G. Caldwell, "Non-parametric imitation learning of robot motor skills," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 5266–5272.

[9] Y. Huang and D. G. Caldwell, "A linearly constrained nonparametric framework for imitation learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4400–4406.

[10] J. Zhang and K. Cho, "Query-efficient imitation learning for end-to-end simulated driving," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[11] K. Menda, K. Driggs-Campbell, and M. J. Kochenderfer, "Ensembledagger: A bayesian approach to safe imitation learning," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 5041–5048.

[12] K. Wang, G. Pleiss, J. Gardner, S. Tyree, K. Q. Weinberger, and A. G. Wilson, "Exact gaussian processes on a million data points," *Advances in Neural Information Processing Systems*, vol. 32,

pp. 14 648–14 659, 2019.

[13] J. R. Gardner, G. Pleiss, D. S. Bindel, K. Q. Weinberger, and A. G. Wilson, "Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration," in *NeurIPS*, 2018.

[14] A. Grosnit, R. Tutunov, A. M. Maraval, R.-R. Griffiths, A. I. Cowen-Rivers, L. Yang, L. Zhu, W. Lyu, Z. Chen, J. Wang *et al.*, "High-dimensional bayesian optimisation with variational autoencoders and deep metric learning," *arXiv preprint arXiv:2106.03609*, 2021.

[15] M. Watter, J. T. Springenberg, J. Boedecker, and M. Riedmiller, "Embed to control: A locally linear latent dynamics model for control from raw images," *arXiv preprint arXiv:1506.07365*, 2015.

[16] N. Levine, Y. Chow, R. Shu, A. Li, M. Ghavamzadeh, and H. Bui, "Prediction, consistency, curvature: Representation learning for locally-linear control," *arXiv preprint arXiv:1909.01506*, 2019.