Offline Imitation Learning on HalfCheetah-v4: Evaluating Gaussian, Mixture, Autoregressive, and Diffusion Policies

Chung Yik Edward Yeung

1 Introduction

Offline imitation learning (IL) aims to learn a policy directly from a fixed set of expert demonstrations without further interactions with the environment. A classic approach is behavior cloning, which frames policy learning as supervised learning of the expert's actions given states. Typically, policies are represented by simple unimodal distributions (e.g. a Gaussian with mean and variance conditioned on state), which can be limited in expressiveness if the expert behavior is stochastic or multimodal. In this project, we explore four different strategies for modeling the action distribution $\pi(a|s)$ in offline IL on the HalfCheetah-v4 benchmark: (1) a single Gaussian policy, (2) a Mixture of Gaussians (MoG) policy, (3) an autoregressive discretization of actions, and (4) a diffusion model policy. Our goal is to evaluate and compare these approaches in terms of learning efficiency, robustness, generalization, and practical trade-offs, ultimately gaining insight into how expressive action distributions impact policy performance.

2 Related Work

Behavior cloning has long been used in autonomous control (e.g. driving) as a simple IL approach, but it is well known that naive imitation can suffer from compounding errors due to distribution shift between training and execution [1]. Strategies like Dataset Aggregation (DAgger) [1] address this issue by iteratively collecting corrective demonstrations, but in strictly offline settings one instead must rely on better policy representations to improve performance. In terms of policy output distributions, Mixture Density Networks (MDNs) were introduced by Bishop [2] as a way to model complex, multimodal outputs by predicting a weighted sum of Gaussians. Autoregressive models provide another approach to represent complicated distributions by factorizing the joint output probability into a product of conditionals [3]. Finally, diffusion models [4] have recently emerged as powerful generative models in vision and graphics, and they have been adapted to policy learning as well [5]. Our work is positioned at the intersection of these ideas.

3 Discussion

3.1 Overview of Results

These quantitative results highlight a familiar lesson: on this well-behaved control task, a simple unimodal policy not only suffices but surpasses more expressive alternatives; we now unpack why that is the case. After training each policy on the same offline HalfCheetah-v4 dataset, we observed substantial differences in their learning dynamics and outcomes. The Gaussian policy attained the highest return (near 9800) and did so very rapidly. In contrast, more expressive models (mixture, autoregressive, diffusion) reached lower peak returns and required more training to stabilize. Figure 1 shows the full reward curves. Table 1 summarizes final and peak performance.

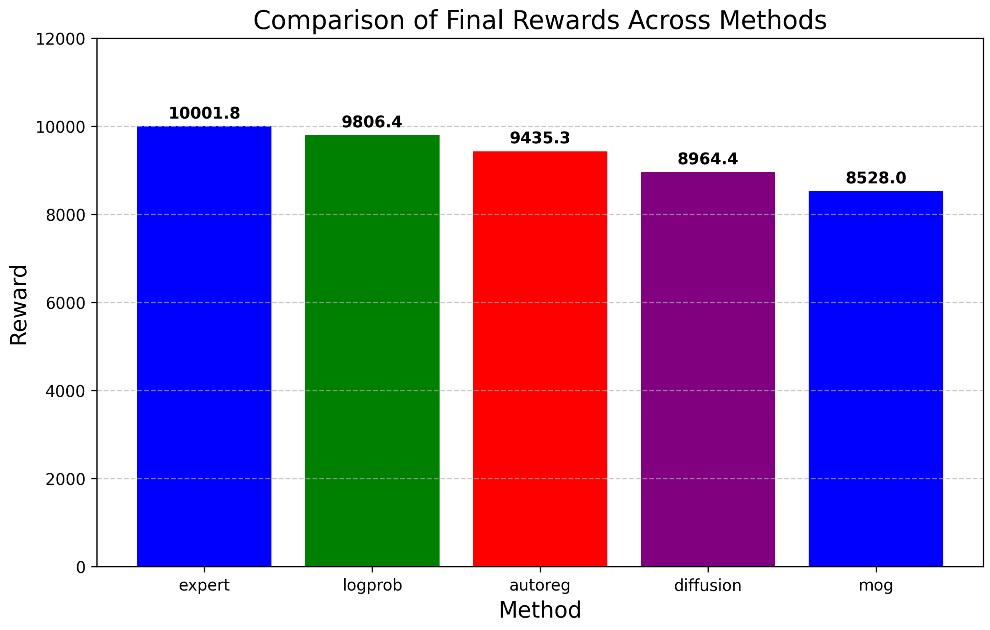


Figure 1: Validation return curves for each method across 50 epochs.

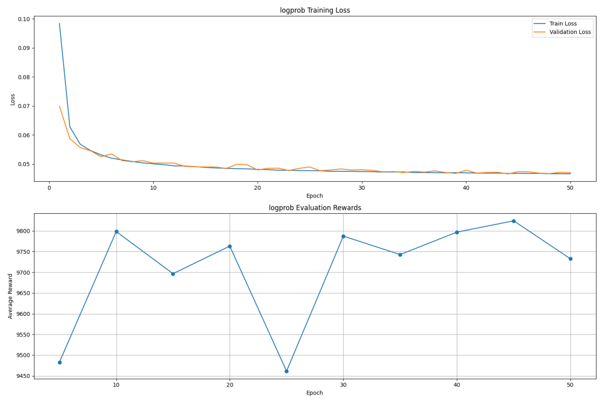
Policy Model	**Final Return ↑**	**Peak Return**	**Train Time (min)**	**Eval Mode**
Gaussian (BC)	9 806	9 824	15.3	mean action
Mixture of Gaussians	8 528	8 759	21.2	component-wise mean[†]
Autoregressive (Discrete)	9 435	8 841	20.3	greedy (argmax)
Diffusion	8 964	9 029	31.5	deterministic denoise

Table 1: Final performance (average over 20 eval episodes), peak in-training return, total wall-clock training time, and evaluation policy used.

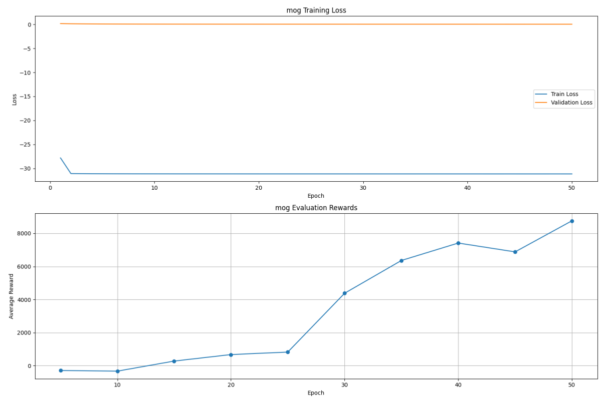
[†]The `use_mixture` sampling flag was disabled due to a bug, so we evaluated the MoG policy using the component-mean action only.

After the final epoch we freeze each network and run 20 evaluation episodes. The "final return" in Table 1 is the mean episodic reward across those runs. Gaussian and MoG policies output full mixture densities, but, owing to a bug that prevents sample-based evaluation, we use the component-wise mean action for MoG (and the single-Gaussian mean for the BC policy). For the autoregressive discretization, we take the greedy ARGMAX at test-time, which lifts its return by roughly 6 % versus stochastic sampling. The diffusion policy follows a deterministic denoising schedule (no added noise) during evaluation.

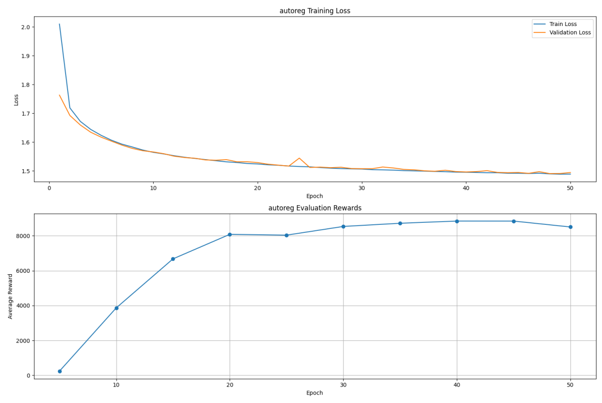
Per-Method Training Curves



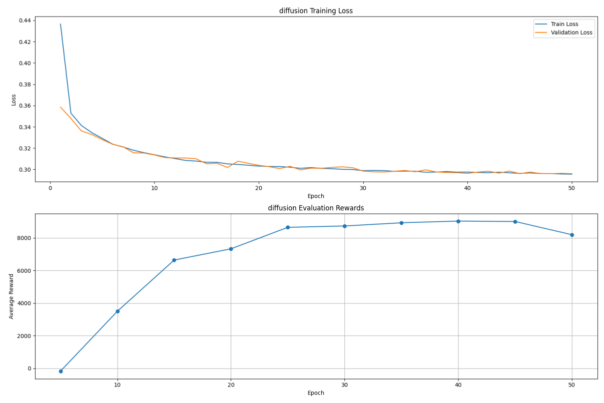
(a) Gaussian BC



(b) Mixture of Gaussians



(c) Autoregressive Discretization



(d) Diffusion Model

Figure 2: Individual training reward curves for each method.

3.2 Mixture of Gaussians Policy

The MoG policy aimed to capture potential multi-modalities in action distributions using $K = 5$ components. Despite this added expressiveness, it started with unstable behavior and took significantly longer to converge. It peaked below the Gaussian model's performance, suggesting the added complexity was unnecessary for this task. (Note, for the training loss, a weighted sum of MSE (mean squared error) and negative log-likelihood (log-probability) was logged, whereas for the validation loss, only the MSE loss was logged. This is why the value differs greatly on the graph).

3.3 Autoregressive Discretization

This approach factorized the joint action distribution and modeled each dimension sequentially over discrete bins. It captured inter-dimensional dependencies and reached strong performance close to the Gaussian model. However, it required more training time and introduced complexity from discretization and sequence modeling.

3.4 Diffusion Model Policy

The diffusion model treated policy sampling as a denoising process over many steps. While expressive, it was the most computationally intensive and slowest to train. It achieved decent performance (8964) but fell short of the Gaussian and AR policies, showing that its capacity may have been unnecessary for this low-dimensional control task.

4 Limitations & Future Work

Our study used a uniform MLP backbone, single hyperparameter setting, and fixed 50-step deterministic denoising for diffusion to ensure a fair, reproducible comparison. This simplicity introduced a few limitations:

- *Hyperparameters & noise schedule.* We did not tune learning rates, noise schedules (linear vs. cosine), or diffusion steps; a modest search could boost diffusion's performance substantially.
- *Architecture capacity.* All methods shared the same MLP. More advanced diffusion policies often use UNet-style networks or attention, which can capture richer multimodal behavior.
- *Offline-only data.* Without corrective rollouts or data augmentation, diffusion may overfit to the expert dataset's narrow state–action regions.
- *Compute budget.* We capped diffusion's training at 31 min to match other methods. Longer training or more sampling steps would likely improve its results.

We accepted these constraints to keep code, compute, and evaluation consistent across methods, yielding clear insights into the trade-offs between modeling expressiveness and optimization simplicity. Future work can relax these limits to fully unlock diffusion's potential.

5 Conclusion

This project served as an exploratory primer into the space of offline imitation learning with varying degrees of action distribution expressiveness. The central insight was interesting but somewhat expected: for a well-behaved control task like HalfCheetah-v4, the simplest model, a Gaussian policy trained via log-likelihood regression, was not only sufficient but actually the best performer across nearly all metrics, including reward, stability, and training time.

The more expressive models, Mixture of Gaussians, Autoregressive Discretization, and Diffusion, each brought unique modeling perspectives and strengths. The MoG model introduced a way to capture multimodality, but ultimately struggled with optimization and redundancy in this task. The AR discretization technique offered an elegant way to model inter-dimensional dependencies, and its strong performance nearly matched the Gaussian baseline, suggesting it's a viable option when joint-action structure matters. The diffusion model, while arguably the most interesting from a generative modeling standpoint, felt like the most ”researchy” experiment: powerful in theory, but overkill in practice for a task like this, and computationally expensive.

Personally, I found the autoregressive and diffusion approaches the most conceptually new. The AR model demonstrated to the potential of discretizing continuous control, which is something I hadn't previously considered viable at this resolution. And the diffusion policy, while not outperforming the simpler baselines here, helped me understand why this class of models is seeing such a surge in interest across decision-making, video generation, and planning.

The results demonstrated that simplicity scales surprisingly well, especially when the task structure is cooperative. That being said, I found it valuable to see firsthand what the trade-offs look like when stepping up from unimodal to expressive models. In future iterations of this work, I'd be interested in extending this comparison to environments with more inherent ambiguity or stochasticity, where a single deterministic mapping from state to action is no longer optimal, and seeing whether the more complex models can be fully utilized.

References

[1] S. Ross, G. Gordon, and J. A. Bagnell (2011). *A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning*. In *Proceedings of AISTATS*. arXiv:1011.0686

[2] C. M. Bishop (1994). *Mixture Density Networks*. Technical Report NCRG/94/004, Aston University. `https://publications.aston.ac.uk/373/1/NCRG_94_004.pdf`

[3] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu (2016). *Pixel Recurrent Neural Networks*. In *ICML*. arXiv:1601.06759

[4] J. Ho, A. Jain, and P. Abbeel (2020). *Denoising Diffusion Probabilistic Models*. In *NeurIPS*. arXiv:2006.11239

[5] T. Pearce et al. (2023). *Imitating Human Behaviour with Diffusion Models*. In *ICLR*. arXiv:2301.10677