In this Supplemental Material, we provide further details, experiments, and descriptions of the attached media, to reinforce the results and conclusions from the main body of our paper. For a more fluid viewing experience please look through our project website, where videos (and corresponding descriptions) are side-by-side: https://sites.google.com/view/eliciting-demos-corl22/home.

# A Additional Dataset Details

**`Square Nut`** [1]. The state space for this task is similar to Mandlekar et al. [1]. We use a proprioceptive state consisting of the robot's end-effector position (3-DoF), end-effector rotation as a quaternion (4-DoF), the gripper position (2-DoF) and a coordinate-based state representation for encoding object positions and poses (14-dim). We use the data as it was originally collected by Mandlekar et al. [1], using a 3DConnexion SpaceMouse for 6-DoF teleoperation. The horizon is set to 500 steps.

We randomly sample 50 demos from the proficient operator [1] to initialize the base dataset. Operators 1 through 4 are Better OP 1, Better OP 2, Okay OP 1 and Okay OP 2 from the Robomimic multi-human dataset.

**`Round Nut`** [1, 2]. The state space for this task is the same as Hoque et al. [2]; complete robot proprioception states and object states are included. The data was collected using a keyboard. The horizon is set to 400 steps.

**`Hammer Placement`** [3]. The state space for this task is the complete robot proprioception state and object. The base demonstrations include 20 demonstration collected by the proficient demonstrator and 5 demonstrations collected by the demonstrator using interactive interventions like in Kelly et al. [4]. These interactive on-policy demos help us in learning a decent base policy. Data was collected with a keyboard. The horizon is set to 175 steps.

# B Policy Training & Other Implementation Details

**Architecture.** We train an ensemble of 5-MLPs. Each MLP has 2 hidden layers, a hidden size of 1024. We use ReLU activations, LayerNorm [5] and a dropout of 0.5 [6] between the hidden layers.

**Training.** We train the models using an ADAM [7] optimizer with a learning rate of 1e-3 for 1000 epochs with a batch size of 512. The models are trained to reduce the mean squared error between the ground truth actions and the predicted actions.

**Evaluation.** The models are evaluated for 50 rollouts for their respective maximum horizons or till the task is completed. Checkpoints are evaluated at every 200 epochs. We also evaluate the checkpoint with the best validation loss.

**Thresholds for compatibility.** We use the thresholds detailed in Table 1 to compute the compatibility score $\mathcal{M}$ for the new demonstrations $\mathcal{D}_{\text{new}}$. Likelihood is measured using a negative mean squared error between the actions predicted by $\pi_{\text{base}}$ and the provided actions $a_{\text{new}}$. The novelty of a state is measured by the standard deviation in the predicted actions from the ensemble policy. To select these thresholds, we assume access to a compatible and an incompatible trajectory in addition to the base demonstrations. We regress these thresholds based on a 2D compatibility map of likelihood vs novelty.

| Parameter | `Square Nut` | `Round Nut` | `Hammer Placement` |
|---|---|---|---|
| Novelty $\eta$ | 0.05 | 0.05 | 0.06 |
| Likelihood $\lambda$ | 0.4 | 0.35 | 0.35 |

Table 1: Thresholds for novelty (std of predicted actions) and likelihood (mean squared error between predicted actions and provided actions). The standard deviation and the MSE of actions were averaged across the dimensions of the action space.

# C Baseline Results

| Operator | Round Nut | | | Hammer Placement | | |
|---|---|---|---|---|---|---|
| | 5-MLP | MDN | RNN | 5-MLP | MDN | RNN |
| Base | 13.3 (2.3) | 8.0 (4.0) | 14.7 (2.3) | 24.7 (6.1) | 11.3 (1.2) | 43.3 (13.3) |
| Operator 1 | 26.7 (11.7) | 29.3 (9.5) | 31.3 (8.3) | 38.0 (2.0) | 30.7 (15.5) | 30.0 (8.0) |
| Operator 2 | 22.0 (7.2) | 11.3 (3.1) | 15.3 (3.1) | 33.3 (3.1) | 12.0 (3.5) | 24.7 (3.1) |
| Operator 3 | 17.3 (4.6) | 10.7 (7.6) | 4.7 (3.1) | 8.0 (0.0) | 12.0 (5.3) | *48.0 (15.6)* |
| Operator 4 | 7.3 (4.6) | 4.7 (3.1) | 13.3 (2.3) | 4.0 (0.0) | 6.7 (2.3) | 8.7 (5.0) |

Table 2: Success rates on `Round Nut` and `Hammer Placement` (mean/std across 3 training runs) for policies trained on $\mathcal{D}_{\text{new}}$ from different operators using different models.

| Operator | Square Nut | | |
|---|---|---|---|
| | 5-MLP | MDN | RNN |
| Base | 38.7 (2.1) | 23.3 (1.2) | 30.7 (1.2) |
| Operator 1 | 54.3 (1.5) | 27.3 (8.3) | 31.3 (1.2) |
| Operator 2 | 40.3 (5.1) | 15.3 (6.1) | 10.7 (1.2) |
| Operator 3 | 37.3 (2.1) | 12.0 (2.0) | 10.0 (2.0) |
| Operator 4 | 27.3 (3.5) | 10.0 (0.0) | 10.7 (3.1) |

Table 3: Success rates on `Square Nut` (mean/std across 3 training runs) for policies trained on $\mathcal{D}_{\text{new}}$ from different operators using different models.

## C.1 Mixture Density Network (MDN)

**Architecture.** We train a Mixture Density Network (MDN) with 2 components corresponding to the 2 operators in the aggregated dataset $\mathcal{D}_{\text{base}} \cup \mathcal{D}_{\text{new}}$. The MDN is modelled as an MLP with 2 hidden layers and a hidden size of 1024. We use ReLU activations, LayerNorm [5] and a dropout of 0.5 [6] between the hidden layers.

**Training.** We train the model using an ADAM [7] optimizer with a learning rate of 1e-4 for 1000 epochs with a batch size of 512. The models are trained to maximize the log likelihood of the expert actions.

**Evaluation.** The models are evaluated for 50 rollouts for their respective maximum horizons or till the task is completed. Checkpoints are evaluated at every 200 epochs. We also evaluate the checkpoint with the best validation loss. We use a low-noise evaluation scheme similar to Mandlekar et al. [1], setting the scale of the Gaussian components to 1e-4 during the evaluation phase.

**Results and Discussion.** From the results in Table 2 and Table 3, we find that using an MDN is worse, in general, compared to an ensemble of MLPs. The trends of operators being compatible to varying degrees with the base dataset holds even when using an MDN. Further, when we aggregate demonstrations from multiple users, it is difficult to pre-define the number of modes (one mode per user) for the MDN. Thus, trying to model multiple modes using an MDN does not help mitigate the lack of compatibility between $\mathcal{D}_{\text{base}}$ and $\mathcal{D}_{\text{new}}$. We also find that the uncertainty estimates in the MDN are not calibrated and tend to collapse to a constant value, making it difficult to use for active elicitation (as we have no metric to tell novel states apart from familiar ones).

## C.2 Recurrent Neural Network (RNN)

**Architecture.** We train an ensemble of 5 LSTM [8] models with two layers and a hidden size of 512.

**Training.** We train the models using an ADAM [7] optimizer with a learning rate of 1e-3 for 1000 epochs with a batch size of 512. The models are trained to reduce the mean squared error between the ground truth actions and the predicted actions.

**Evaluation.** The models are evaluated for 50 rollouts for their respective maximum horizons or till the task is completed. Checkpoints are evaluated at every 200 epochs. We also evaluate the checkpoint with the best validation loss.

**Results and Discussion.** From the results in Table 2 and Table 3, we find that the ensemble of RNNs, in general, is comparable to an ensemble of MLPs. Similar to an ensemble of MLPs and MDNs, the trends are quite consistent, albeit a couple of exceptions (e.g., Operator 3 in `Hammer Placement`). The added computational load of using a sequential model does not mitigate the problem of incompatibility between the demonstrations from different users. We prefer to use an ensemble of MLPs (for their lower computational load) to test the validity of our compatibility metric and demonstrations elicitation procedure. Our procedure can easily be extended to sequential models.

# D Real Robot Task Details: Food Plating

**Hardware Details.** We use a Franka Emika Panda arm for our experiments. We use a RealSense camera to record visual observations (as RGB images). For control, we predict 7-DoF joint actions and use the Polymetis library [9] for low-level impedance control. We keep the gripper of the Panda arm in a fixed position grasping the pan throughout the task.

**Policy Architecture.** We train an ensemble of 5 visually-conditioned policies. We use a ResNet34 backbone pretrained on Imagenet [10] to encode the visual observations, keeping the ResNet weights frozen. The robot proprioceptive state consists of the end effector position and pose (as a quaternion), concatenated with the visual embeddings and passed through an MLP to predict the actions. The MLP consists of two hidden layers with a hidden size of 64 and GELU [11] activations.

**Active Elicitation.** If a demonstration is rejected, we provide corrective feedback to demonstrators after the demo has been recorded. We show a video of the incompatible parts of the trajectory, retrieve and play the closest expert demo to the rejected one. For the retrieval of corrective demos, we look at the similarity of demos in the state space. This is done by measuring the L2 distance of the ResNet embeddings. This isn't a perfect measure and that lots of other work tries to solve this problem; we choose ResNet features to be expedient.

**Policy Training.** We train the ensemble of visual policies for 20 epochs with a batch size of 512. The model is trained to minimize the mean squared error (MSE) between predicted and recorded actions. We use an AdamW optimizer [12] with a learning rate of 1e-3.

**Evaluation.** For evaluation, we choose five points for the location of the plate and evaluate each policy for 5 rollouts.

# E Additional Results on Active Elicitation

**`Round Nut`**. We collect data from 16 users (age = $23.7 \pm 1.7$, 11 males, 5 females). Each user is either assigned to the naive or informed condition.

**`Hammer Placement`**. We collect data from 4 users (age = $23.2 \pm 0.9$, 4 males). Each user performs the naive condition first and then the informed condition.

**`[Real] Food Plating`**. We collect data from 4 users (age = $23.0 \pm 1.1$, 1 male, 3 females). Each user performs the naive condition first and then the informed condition.

**Post Study Survey.** We asked users to rate their experience in collecting the demonstrations by asking them questions related to mental demand, task difficulty and task comprehension on a 5-point Likert scale. These questions were inspired from Hart [13].

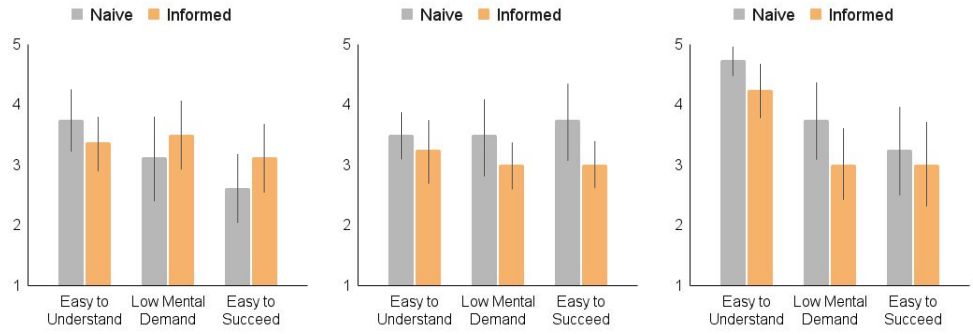


Figure 1: Results of our post-study survey. All responses are collected on a 5-point Likert scale (1: Strongly Disagree, 5: Strongly Agree).

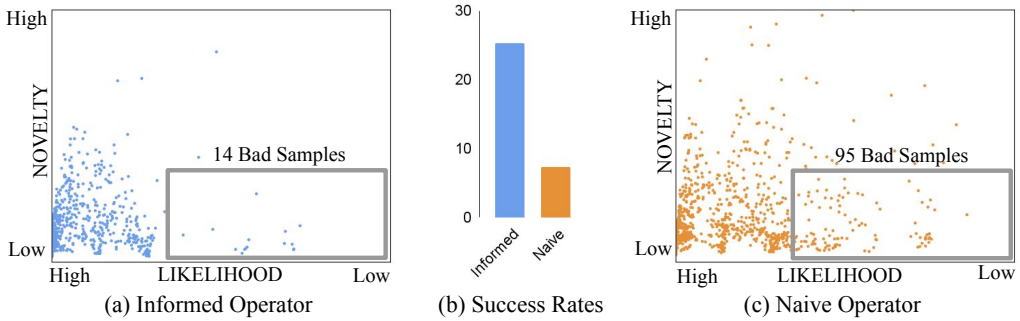


Figure 2: (a) and (c) show 2D "maps" of demonstrations collected from an informed operator and a naive operator respectively. (b) shows the success rates on using the two sets of demonstrations to train a policy.

**Discussion.** From the post-study survey Fig. 1, we find that our method is slightly more difficult to understand (+0.37; averaged across 3 tasks), requires marginally more mental demand (+0.30; averaged across 3 tasks), and is a little more difficult to succeed at (+0.20; averaged across 3 tasks). We find that this marginal increase in difficulty and effort in performing the task lead to the collection of significantly better demonstrations. For instance, we see in Fig. 2 that the informed operator, using our active elicitation interface, is much better at giving more compatible demonstrations that the naive operator. This is reflected in the success rates acheived by the corresponding policies (25.3 v/s 7.3).

**Trajectory Lengths.** In Table 4, we present the average trajectory lengths for demonstrations collected by the base user, naïve users, and informed users. We find that informed users tend to be more optimal in providing demonstrations while also providing demos of a similar style to the base user. For demonstrations with the real food plating task, the base user's style requires longer trajectories on average compared to a naïve user's style. Our active elicitation procedure is able to bring the average trajectory length of an informed user closer to that of the base user. So, we are able to elicit behavior that matches a style, not solely optimizing for shorter trajectories.

# F Active Elicitation with Human-Gated (HG) DAgger

**Procedure.** We use the same interface as described in Section 5 to collect demonstrations interactively using Human-Gated DAgger [4]. Users were asked to help a robot complete the **`Round Nut`** task successfully five times. They were instructed to intervene and help the robot when they thought the

| Task | Base | Naïve | Informed |
|---|---|---|---|
| `Round Nut` | 87.3 | 95.9 (12.5) | 88.9 (6.8) |
| `Hammer Placement` | 174.6 | 185.5 (34.3) | 174 (8.7) |
| `Real: Food Plating` | 306.5 | 263.7 (10.26) | 278.3 (8.9) |

Table 4: Average trajectory lengths for demonstrations collected using active elicitation and naïve collection.

| Operator | Naïve | Informed |
|---|---|---|
| Base | 13.3 (2.3) | - |
| Operator 1 | 24 (3.5) | 25.3 (5.0) |
| Operator 2 | 18 (7.2) | 23.3 (4.2) |
| Operator 3 | 31.3 (9.9) | 21.3 (2.3) |
| Operator 4 | 29.3 (5.8) | 32.7 (7.0) |

Table 5: Success rates (mean/std across 3 random seeds) for user studies evaluating both naive and informed demonstration collection using HG-DAgger against base users for the `Round Nut` task.

robot was stuck or was making a mistake in completing the task. They were also told to give control back to the robot when they thought the robot could complete the task successfully.

$\mathcal{D}_{\text{base}}$ consists of 30 trajectories collected by a proficient operator. For this task, we perform a longitudinal study with $n = 3$ participants, where users are first asked to complete 5 demonstrations in the naive condition and then 5 demonstrations in the informed condition. This allows us to measure the effect of the interface in eliciting demonstrations within subjects.

**Results and Discussion.** Informed elicitation works better for three out of four operators (see Table 5) but the gains are lower compared to the condition where we collect complete trajectories. Further, we observe that none of the conditions result in a policy that is worse than the base policy. We find that the base policy is quite good and only requires intervention in a few "critical states" like picking the nut up or inserting the nut into the peg. Further, the results also show that the high frequency feedback to the users from our interface does not discourage them from intervening and providing corrections. Our results our limited by the number of users we test in this condition and also by the task that we consider. Future work will address how active elicitation might help in interactive imitation learning across more users and more diverse tasks.

## G Operator-wise Success Rates

## References


[1] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2021.

[2] R. Hoque, A. Balakrishna, E. R. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg. ThriftyDAgger: Budget-aware novelty and risk gating for interactive imitation learning. In *Conference on Robot Learning (CoRL)*, 2021.

[3] Y. Zhu, P. Stone, and Y. Zhu. Bottom-up skill discovery from unsegmented demonstrations for long-horizon robot manipulation. *IEEE Robotics and Automation Letters (RA-L)*, 7:4126–4133, 2022.

| Operator | Success Rates |
|---|---|
| Base | 13.3 (2.3) |
| Naïve 1 | 16 (3.5) |
| Naïve 2 | 7.3 (4.6) |
| Naïve 3 | 6.7 (1.2) |
| Naïve 4 | 8.0 (2.0) |
| Naïve 5 | 13.3 (4.2) |
| Naïve 6 | 7.3 (3.1) |
| Naïve 7 | 4.7 (1.2) |
| Naïve 8 | 13.3 (2.3) |
| Informed 1 | 25.3 (1.2) |
| Informed 2 | 20.0 (2.0) |
| Informed 3 | 18.0 (3.5) |
| Informed 4 | 11.3 (2.3) |
| Informed 5 | 16.0 (3.5) |
| Informed 6 | 5.3 (1.2) |
| Informed 7 | 15.3 (1.2) |
| Informed 8 | 14.0 (3.5) |

Table 6: Success rates (mean/std across 3 random seeds) for different operators on the `Round Nut` Task

| Operator | Naïve | Informed |
|---|---|---|
| Base | 24.7 (6.1) | - |
| Operator 1 | 8.0 (0.0) | 28.0 (6.0) |
| Operator 2 | 33.3 (3.1) | 52.7 (10.1) |
| Operator 3 | 38.0 (2.0) | 35.3 (2.3) |
| Operator 4 | 4.0 (0.0) | 11.3 (2.3) |

Table 7: Success rates (mean/std across 3 random seeds) for different operators on the `Hamemr Placement` Task


[4] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. HG-DAgger: Interactive imitation learning with human experts. In *International Conference on Robotics and Automation (ICRA)*, pages 8077–8083, 2019.

[5] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

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

[7] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

[9] Y. Lin, A. S. Wang, G. Sutanto, A. Rai, and F. Meier. Polymetis. https://facebookresearch.github.io/fairo/polymetis/, 2021.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[11] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[12] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[13] S. G. Hart. NASA task load index (nasa-tlx); 20 years later. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 50, pages 904–908, 2006.