



<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper presented at *IEEE Int. Conf. on Intelligent Robots and Systems*.

Citation for the original published paper:

Marta, D., Holk, S., Pek, C., Tumova, J., Leite, I. (2023)

VARIQuery: VAE Segment-based Active Learning for Query Selection in Preference-based Reinforcement Learning

In:

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-333948>

VARIQuery: VAE Segment-based Active Learning for Query Selection in Preference-based Reinforcement Learning

Daniel Marta*, Simon Holk*, Christian Pek, Jana Tumova, and Iolanda Leite

Abstract—Human-in-the-loop reinforcement learning (RL) methods actively integrate human knowledge to create reward functions for various robotic tasks. Learning from preferences shows promise as alleviates the requirement of demonstrations by querying humans on state-action sequences. However, the limited granularity of sequence-based approaches complicates temporal credit assignment. The amount of human querying is contingent on query quality, as redundant queries result in excessive human involvement. This paper addresses the often-overlooked aspect of query selection, which is closely related to active learning (AL). We propose a novel query selection approach that leverages variational autoencoder (VAE) representations of state sequences. In this manner, we formulate queries that are diverse in nature while simultaneously taking into account reward model estimations. We compare our approach to the current state-of-the-art query selection methods in preference-based RL, and find ours to be either on-par or more sample efficient through extensive benchmarking on simulated environments relevant to robotics. Lastly, we conduct an online study to verify the effectiveness of our query selection approach with real human feedback and examine several metrics related to human effort.

I. INTRODUCTION

Translating complex goals into robot policies has previously been carried out by hand-crafting meticulous reward functions which requires considerable fine-tuning and expert human knowledge. A popular alternative to hand-crafting reward functions devises robot policies to be taught by humans through human-in-the-loop methods [1], [2]. This paper centers on a promising approach that involves learning by iteratively querying humans for preferences between pairs of state-action sequences generated from a policy [3]–[6]. Preferences allow non-experts to contribute without explicitly providing demonstrations, which may be infeasible or expensive to obtain. By learning from human preferences, we can directly convey subtle nuances of the reward function that are difficult to model *a priori* [4].

Despite its potential benefits, preference-based RL poses a significant challenge in terms of scalability due to the considerable amount of human feedback needed to capture complex objectives. While many preference-based algorithms rely on

This research has been carried out as part of the Vinnova Competence Center for Trustworthy Edge Computing Systems and Applications at KTH, and partially supported by the Swedish Foundation for Strategic Research (SSF FFL18-0199) and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

* These authors contributed equally to this work.

All of the authors are with the Division of Robotics, Perception and Learning, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 114 28 Stockholm, Sweden. The authors are also affiliated with Digital Futures. Mail addresses: {dlmarta, sholk, pek2, tumova, iolanda}@kth.se

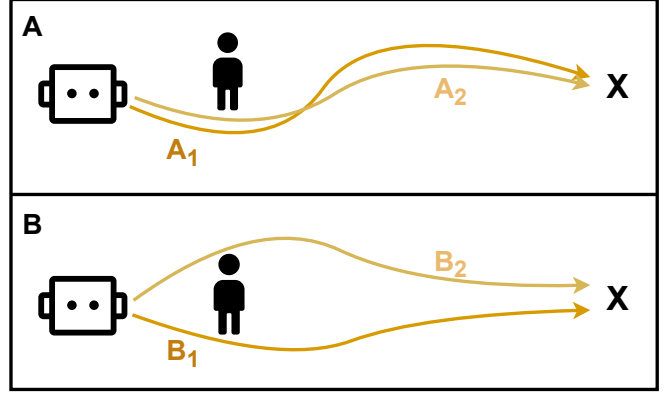


Fig. 1: Example showcasing queries *A* and *B* drawn from different query selection methods. Each query element depicts a trajectory in a robot social navigation scenario.

abundant human feedback to learn from human preferences, the cost of obtaining such feedback can be significant. However, simply limiting the amount of human feedback may result in a biased query set [7] that does not generalize well, necessitating over-querying of humans to overcome this issue. This work focuses on alleviating the burden on human experts by creating sets of queries that are representative of the larger population of state-action sequences produced by a policy. Query selection, which is often overlooked, is a crucial aspect of the preference-based RL process.

To better illustrate the importance and need for more informative methods of selecting queries, we present an example of a social robot navigation scenario. Fig. 1 displays two preference queries with two trajectories each. In query *A* (Fig. 1.a), we observe two trajectories that were obtained through uniform random sampling. Because trajectories A_1 and A_2 are quite similar, a human may struggle to make a preference decision, since they are structurally similar and there is no clear preference. Eliciting preferences from humans incurs a cost, and a human skipping a query altogether, is not only detrimental to learning performance - as a skipped query carries no additional information - but also raises the overall cost. In contrast, in query *B* (Fig. 1.b), trajectories B_1 and B_2 are sampled from different clusters, ensuring greater query diversity. Additionally, it would be advantageous if the selected query *B* is likely to improve our reward model.

Our work proposes a query selection method with the goal of selecting diverse and informative queries. We achieve this through an approach that leverages two synergistic components: (1) latent space representations of query elements provided by a variational autoencoder (VAE) [8]; and (2) ranking of the resulting queries by current reward

model estimations. In this manner, we ensure queries are informative with respect to the current reward model while avoiding similar queries from being repeatedly presented to humans. Our contributions can be summarized as follows:

- 1) **VARIQuery**: VAE Segment-based Active LeaRnIng for **Query** Selection in Preference-based Reinforcement Learning (see Sec. IV), a query selection method which aims at decreasing query redundancy by simultaneously taking into account two factors: the underlying structure of state-action sequences; and their likeliness of improving the reward model.
- 2) We compare VARIQuery to other state-of-the-art query selection methods, currently used in preference-based RL.
- 3) We conduct an online user study in a crowded robot navigation simulated environment with the goal of collecting human preferences from queries produced by our method against several query selection methods and evaluating human effort on quantitative metrics.

II. RELATED WORK

Deep Active Learning. Active learning (AL) techniques attempt to maximize a model’s performance when labelled data is scarce and dependent on human effort. AL has gathered increasing research interest [9], specifically in the computer vision community. A subset of AL techniques referred to as pool-based [9], [10] aims at finding the best samples to be queried, giving an evaluation and ranking of a set of unlabelled samples. To this end, there is a plethora of various query strategies that comprise uncertainty-based approaches [11]–[13], expected model changes approach [14], [15] and diversity-based [16]–[18]. Additionally, numerous studies have explored hybrid query strategies [19]–[21] that aim to strike a balance between uncertainty and diversity of query samples. However, sampling-based solely on uncertainty often leads to sampling bias [7], resulting in selected samples that do not accurately represent the distribution of unlabeled datasets. Deep active learning (DeepAL) [22]–[24] aims at leveraging deep learning to improve previous AL techniques. We turn to DeepAL to address the high-dimensionality of sequences of state-action pairs, while exploring suitable pool-based hybrid techniques, to formulate a query selection method.

More specifically, we take inspiration from works such as VAAL (Variational Adversarial Active Learning) [25] which leverage VAEs (Variational AutoEncoder) [8] to build an adversarial network model to discriminate between unlabeled and labelled data. The purpose is to discriminate between dissimilarities in the latent space to select images for labelling. We gather inspiration from TA-VAAL [26] (Task-Aware Variational Adversarial Active Learning) which aims at setting a balance between uncertainty and diversity. TA-VAAL achieves this balance by integrating a loss prediction module [27] with a ranking approach [28] (RankCGAN). TA-VAAL operates on the premise that uncertainty-based techniques overlook the entire data distribution, while distribution-based

methods disregard the overall task structure. Drawing inspiration from DeepAL techniques, our approach involves contributing with an hybrid query strategy that considers the underlying structure of preference queries in the context of preference-based RL.

Query synthesis in preference-based RL The utilization of preferences for learning has gained a significant amount of attention in the literature [4], and poses as a promising RL approach suitable for robotics [29]. However, the efficacy of preference-based RL as a viable approach is contingent upon the capability to generate informative queries when soliciting feedback from humans on pairs of trajectories.

Initial strategies proposed on elaborating objectives to explore the preference function space [30]. In Akrouir et al. [31], a utility function was defined alongside an exploration term, to derive a diversity function applicable to discrete state spaces. Later on, Akrouir proposes a different selection criterion that maximizes the expected utility of selection [3], [32], similarly to AL techniques. There are several relevant works which also actively select queries by searching a discrete or sampled set [33]–[35]. In an AL approach, Sadigh et al. [36], propose a method to actively synthesize queries with the aim of maximizing volume removal from the distribution of potential rewards. In situations where reducing volume (on a Gaussian process regression framework) would prompt a robot to solicit preferences for highly similar trajectories, alternative strategies based on information gain were explored [37], [38]. Current preference-based RL [5], [39] approaches take advantage of state-of-the-art policy gradient methods which make the computation of expected utility intractable. In Cristiano et al. [5], an ensemble variance ranking method is employed and in PEBBLE [39], a particle-based entropy strategy is introduced. Our work differs from the above by simultaneously utilizing latent space representations of queries and their impact on the performance of the reward model.

III. BACKGROUND

A. Learning from Human preferences

A robotic agent operates under a policy, which generates trajectories. Each trajectory can be represented as a sequence of segments $\tau = (\sigma^1, \dots, \sigma^N)$, where N is the number of segments in a trajectory. Trajectory segments consist of sequences of state-action pairs, denoted as $\sigma^j = ((s_t^j, a_t^j), \dots, (s_{t+l}^j, a_{t+l}^j))$, where j denotes the index of the segment, which contains state-action pairs ranging from t to $t + l$, where $l + 1$ is the length of the segment. The goal of preference-based RL [4], [5] is to obtain feedback from humans on segments of state-action pairs, instead of individual pairs, reducing the number of requests. The feedback is gathered by presenting pairwise comparisons of segments to humans, where preferences between two segments, are denoted by a preference operator \succ . Thus, if σ^1 is preferred over σ^2 , we represent it by writing $\sigma^1 \succ \sigma^2$. The resulting preference provided by the human is a 2-D tuple denoted by $\zeta = (\zeta_1, \zeta_2)$, where $\zeta = (1, 0)$ if σ^1 is preferred over σ^2 , and $(0, 1)$ if σ^2 is preferred over σ^1 . If no

preference is observed, the feedback is denoted by $(0.5, 0.5)$, and $(0, 0)$ if unrelated. The aim is to learn a policy π that is according to these preferences. Therefore, preferences need to be mapped to concrete effects in a reward function represented by $\mathcal{R}(s_t, a_t)$. If a human prefers $\sigma^1 \succ \sigma^2$ then we assume $\sum_t \mathcal{R}(s_t^1, a_t^1) > \sum_t \mathcal{R}(s_t^2, a_t^2)$. To achieve this, we estimate \mathcal{R} with $\hat{\mathcal{R}}$. We use a combination of softmax functions to compute the probability of a human choosing one segment over another. The probability is represented as follows:

$$\mu(\sigma^1 \succ \sigma^2) = \frac{\exp(\sum_t \hat{\mathcal{R}}(s_t^1, a_t^1))}{\exp(\sum_t \hat{\mathcal{R}}(s_t^1, a_t^1)) + \exp(\sum_t \hat{\mathcal{R}}(s_t^2, a_t^2))} \quad (1)$$

Preferences are collected from humans and stored alongside segments to form queries, which are tuples of the form $q = (\sigma^1, \sigma^2, \zeta)$. In turn, queries are stored in a query dataset \mathcal{D}_q . We sample mini-batches from \mathcal{D}_q , to compute the loss for the reward estimator $\hat{\mathcal{R}}$. The loss function is defined as the cross-entropy between the predicted preference and the actual preference. Specifically, we use the following equation [5]:

$$\mathcal{L}(\hat{\mathcal{R}}) = - \sum_{(\sigma^1, \sigma^2, \zeta)} \zeta_1 \log \mu(\sigma^1 \succ \sigma^2) + \zeta_2 \log \mu(\sigma^2 \succ \sigma^1) \quad (2)$$

By using Eq. 2, we can train our model to predict preferences accurately to compute the reward and update policy π accordingly.

B. Query selection methods in preference-based RL

Uniform sampling. The simplest query-selection strategy, where a large number of segments σ are sampled uniformly at random to form random queries.

Ensemble-based sampling. Let $\hat{\mathcal{R}} = (\hat{\mathcal{R}}_1, \dots, \hat{\mathcal{R}}_n)$ be an ensemble of n reward estimators, each of which is a function that takes a state-action pair and produces a reward prediction as output. The reward estimators are initialized with different network parameters and trained on the same queries. Let X be a random variable representing the sum of predicted rewards for a given segment, as produced by the ensemble. Then, the variance of the ensemble is given by:

$$\text{Var}(X) = \frac{1}{n} \sum_{i=1}^n \left(\sum_t \hat{\mathcal{R}}_i(s_t, a_t) - \bar{\mu} \right)^2 \quad (3)$$

where $\hat{\mathcal{R}}_i(s_t, a_t)$ is the reward prediction produced by the i -th estimator for the pair (s_t, a_t) , and $\bar{\mu}$ is the mean reward prediction across all estimators in the ensemble, i.e.,

$$\bar{\mu} = \frac{1}{n} \sum_{i=1}^n \sum_t \hat{\mathcal{R}}_i(s_t, a_t) \quad (4)$$

The variance of the ensemble is the average squared deviation of the reward predictions from the mean prediction, divided by the size of the ensemble. The authors of [5] used this variance to rank segments by disagreement across the ensemble members. Then the top ranked segments are sampled to produce queries to be presented to humans.

Particle-based entropy sampling. Given a state-action sequence $\sigma^j = ((s_t^j, a_t^j), \dots, (s_{t+l}^j, a_{t+l}^j))$, let us denote $s^{\sigma^j} = (s_t^j, \dots, s_{t+l}^j)$ its corresponding state sequence. In the particle-based entropy sampling approach, the entropy of a segment σ^j is equivalent to the entropy of its corresponding state sequence $\mathcal{H}(s^{\sigma^j})$ which is approximated via a simplified version of a particle-based entropy estimator $\hat{\mathcal{H}}_\sigma$ [39]:

$$\hat{\mathcal{H}}_\sigma(s^{\sigma^j}) \propto \sum_i^N \log(\|s_i^{\sigma^j} - k_{s_i^{\sigma^j}}\|), \quad (5)$$

where $k_{s_i^{\sigma^j}}$ represents the k -th nearest neighbor (k -NN) of $s_i^{\sigma^j}$. Thus, the entropy of a state $s_i^{\sigma^j}$ is increased by maximizing the distance between the state and the nearest neighbour. An interpretation is to consider, a state as a particle [40] which contributes to the overall entropy of the sequence.

Moreover, queries are also evaluated by their information entropy regarding the predictions of the reward model $\hat{\mathcal{R}}$. More concretely, for each query the information entropy [9] is defined as follows:

$$\mathcal{H}_{\hat{\mathcal{R}}}(\sigma^1, \sigma^2) = - \mu(\sigma^1 \succ \sigma^2) \log_2 \mu(\sigma^2 \succ \sigma^1) - (1 - \mu(\sigma^2 \succ \sigma^1)) \log_2 (1 - \mu(\sigma^1 \succ \sigma^2)). \quad (6)$$

This entropy is used to select queries at the boundary of the decision of $\hat{\mathcal{R}}$. This is analogous to margin sampling [24], [41] in AL, where samples are chosen to be selected for labelling by the difference (margin) of the highest two predicted labels. The smaller the margin M , the greater the uncertainty of a sample. Thus, in particle-based entropy sampling, segment pairs are randomly sampled and ordered according to $\mathcal{H}_{\hat{\mathcal{R}}}$, and the top pairs which have higher information entropy are sampled again and ordered by the maximum entropy of their segments $\hat{\mathcal{H}}_\sigma$.

IV. VARIQUERY

A natural criteria for query selection would be to maximize the expected value of information (EVOI) [42], i.e, selecting queries which will improve policy π with respect to the utility of queries. However there are two limitations which make the computation of EVOI infeasible: (1) The computation of EVOI is intractable as it requires taking an expectation across all conceivable trajectories that are generated by an updated policy π ; (2) It requires the assumption of an utility function [3], [42] for the user (humans), which could potentially introduce a bias and is task dependant. Several approximations have been explored such as sampling queries based on the variance of an ensemble or based on the total entropy of segments (see Sec. III-B).

A. Selecting queries with VARIQuery

In this work we propose a query strategy, which takes into account both the likeliness of improving the reward model $\hat{\mathcal{R}}$, while ensuring query diversity, both relevant when eliciting preferences from humans. Our query selection approach (see Alg. 1 and Fig. 2) contemplates three major steps:

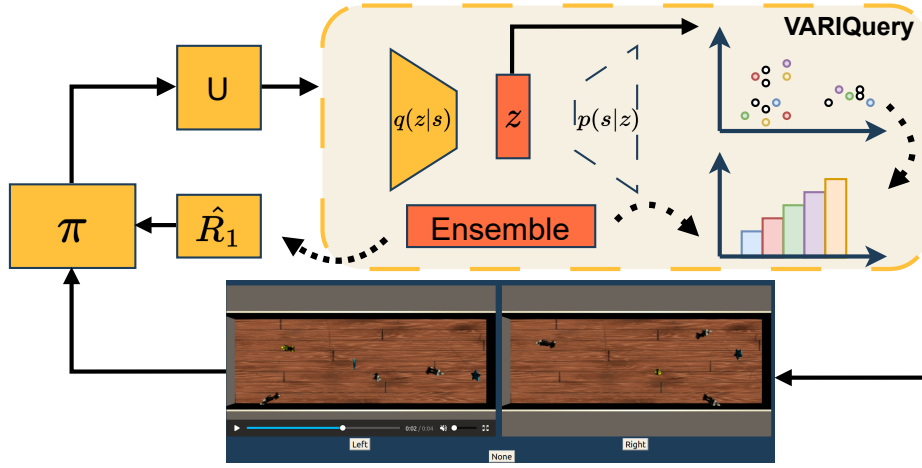


Fig. 2: The VARIQuery framework. The robot collects a set of trajectory segments and processes them into their latent representation. The segments are classified into clusters based on their representations using k -NN and are then collected from distinct clusters to form queries. The queries are then ranked based on the ensemble disagreement. Finally, the chosen queries are shown to and labelled by humans.

- Step 1: We collect a large amount of unlabelled segments \mathcal{D}_{Un} following policy π .
- Step 2: From \mathcal{D}_{Un} , we train a VAE according to Eq. 7. Then we transform each segment into a latent space representation to form \mathcal{D}_z . Both datasets are pair-wise linked, e.g, each segment in \mathcal{D}_{Un} has a latent representation on \mathcal{D}_z (see Sec. IV-B).
- Step 3: We process \mathcal{D}_z into K clusters using k -NN clusters (see Sec. IV-C).
- Step 4: We finally obtain queries by sampling segments uniformly from diverse clusters and ranking them based on the variance of a reward ensemble (see Sec. IV-D).

B. Segment-based VAE Representations

When employing particle-based entropy sampling (see Sec. III-B), one of its constraints is that segments are assessed state by state, without considering the segment's overall structure. For example, consider two segments σ^{B_1} and σ^{B_2} . Both segments may have been selected by having comparable high entropy when summed over states according to Eq. 5, i.e., $\mathcal{H}_\sigma(\sigma^{B_1}) \approx \mathcal{H}_\sigma(\sigma^{B_2})$. However, high entropy alone is a not sufficient criteria to unbiased both segments: segments may have similar entropy because they are structurally similar. Thus, a particle-based entropy approach in isolation may potentially generate redundant queries.

We aim to enhance the analysis and selection of queries to reduce redundancy and increase their informativeness, with the goal of minimizing the human effort involved in the process. To achieve this, we employ a Variational Autoencoder (VAE) [8] to learn a concise and informative representation of state sequences. Specifically, we use an encoder $q(z|s^\sigma)$ to obtain a latent representation $z \in \mathcal{Z}$ from the state sequence s^σ , which captures the underlying structure of the sequence and facilitates query creation. In addition, we utilize a decoder $p(s^\sigma|z)$ to map the learned representation back to the input space. By jointly optimizing the encoder and decoder, we minimize the following loss function:

$$L_{VAE} = -\mathbb{E}_{q(z|s^\sigma)} [\log p(s^\sigma|z)] + KL(q(z|s^\sigma)||p(s^\sigma)) \quad (7)$$

The first term in the loss function measures the reconstruction error, while the second term acts as a regularizer, encouraging the posterior distribution to remain close to the prior distribution $p(s^\sigma)$.

C. Segment-based Clustering

Given the learned representation over state sequences, we utilize the fact that the latent representation forms natural clusters where similar sequences will be closer together and other sequences drift further away in the latent space. Furthermore, the latent representation greatly reduces the dimensionality of the data which makes it more suitable for clustering methods such as k -NN which are sensitive to the *curse of dimensionality* [43]. We therefore perform k -NN to find the clusters and use them to classify state-sequences. Let $\mathcal{D}_{\text{Un}} = \{s^{\sigma^1}, s^{\sigma^2}, \dots, s^{\sigma^n}\}$ be the set of state sequences for different segments in the original feature space, where $s^{\sigma^i} \in \mathbb{R}^k$ represents the i -th state sequence, and $l + 1$ the length of a segment. We first map each state sequence s^{σ^i} to its corresponding latent representation z_i using the encoder $q(z|s^{\sigma^i})$. We then use k -NN to cluster the latent representations into k clusters. For each representation z_i , we find its k nearest neighbors in the latent space, and assign z_i to the cluster that contains the majority of its neighbors. The result is a set of k clusters $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$, where each cluster C_j contains the set of representations $z_i | z_i \in C_j$. Each latent representation z_i is mapped to a state sequence s^{σ^i} by its index i .

D. Ranking queries in VARIQuery

After clustering the state sequences into k clusters $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$ based on their learned latent representations z_i using k -NN, query selection can be performed in two distinct steps. First, to ensure diversity within the queries, trajectory segments are sampled from distinct clusters. Let C_a and C_b be two different clusters selected at random from \mathcal{C} . Then, a segment is sampled uniformly at random from each of these clusters to form a query pair $q = (\sigma^a, \sigma^b)$. The indexes of the sampled latent space representations are used

Algorithm 1: VARIQuery

```
1 Input:  $\pi$  current policy,  $N$  total number of
   unlabelled segments  $\sigma$ ,  $L$  length of segments  $\sigma$ ,  $Q$ 
   number of queries,  $K$  number of  $k$ -NN clusters,  $\hat{\mathcal{R}}$ 
   ensemble of reward networks;
2 Output:  $\mathcal{D}'_q$  dataset of selected queries
3  $\mathcal{D}_q \leftarrow \emptyset$ ;  $\mathcal{D}_z \leftarrow \emptyset$ ;  $\mathcal{D}_{\text{Un}} \leftarrow \emptyset$ ;
   // Step 1 (see Sec. IV-A)
4 for  $i$  to  $N$  do
5   | Sample  $\sigma^i$ , by getting  $s_{t+1}$  following  $a_t \sim \pi(s_t)$ 
6   |  $\sigma^i = \text{sampleSegment}(\pi, L)$ 
7   |  $\mathcal{D}_{\text{Un}} \leftarrow \mathcal{D}_{\text{Un}} \cup \sigma^i$ 
   // Step 2 (see Sec. IV-B)
8 Train VAE from  $\mathcal{D}_{\text{Un}}$  with respect to Eq.7
9  $\text{VAE} = \text{trainVAE}(\mathcal{D}_{\text{Un}})$ 
10 for  $i$  to  $N$  do
11   | Get  $\sigma^i$  from  $\mathcal{D}_{\text{Un}}$ 
12   |  $z_i = \text{VAE}_{\text{encoder}}(\sigma^i)$ 
13   |  $\mathcal{D}_z \leftarrow \mathcal{D}_z \cup z_i$ 
   // Step 3 (see Sec. IV-C)
   // Store  $K$  clusters in  $C$ 
14  $C \leftarrow k\text{-NN}(\mathcal{D}_z, K)$ ;
   // Step 4 (see Sec. IV-D)
15 Sample pairs at random  $\sigma^1, \sigma^2 \sim C$ 
16  $\mathcal{D}_q = \text{samplePairs}(C)$ 
17 Order  $\mathcal{D}_q$  by ensemble variance  $\hat{\mathcal{R}}$  with Eq. 3
18  $\mathcal{D}_{\text{ranked}} = \text{Rank}(\mathcal{D}_q, \hat{\mathcal{R}})$ 
19  $\mathcal{D}'_q = \text{sampleFromTop}(\mathcal{D}_{\text{ranked}})$ 
20 return  $\mathcal{D}'_q$ 
```

to obtain the corresponding segments. Each query pair q is stored in the set of query pairs \mathcal{D}_q . Finally, the queries are ranked based on the variance across the ensemble members using the method described in Sec. III, where the variance of the query is the maximum of the variance of the segments of the query. Thus, \mathcal{D}_q is ordered by ensemble disagreement to obtain the sorted set of query pairs \mathcal{D}'_q . The resulting queries \mathcal{D}'_q are then presented to humans for preference collection. This hybrid approach aims to create queries that are diverse enough so that a human can clearly distinguish between the trajectories while also collecting queries which are likely to impact the reward model the most.

V. EXPERIMENTAL RESULTS

This section involves benchmarking our method across various robotic tasks and query selection methods, with a focus on addressing three essential questions:

- Question 1 (Q1): How does VARIQuery compare to other state-of-the-art query selection methods in terms of sample query complexity and learning performance? (see Sec. V-B)
- Question 2 (Q2): Is the k -NN clustering of the latent space produced by VARIQuery sufficiently able to produce queries with diverse segments? (see Sec. V-C)

- Question 3 (Q3): Can VARIQuery reduce human effort, as measured by the time spent per query and the occurrence of deadlocks during the labeling of query batches? (see Sec. VI)

A. Benchmark details

To verify VARIQuery, we selected two robotics-related control tasks simulated on MuJoCo and available on OpenAI Gym [44]: Walker2d and Cheetah. Using a state-of-the-art policy gradient algorithm, PPO [45], we implement preference-based RL as described in the work by Christiano et al. [5]. In order to eliminate the influence of suboptimal hyperparameters causing randomness in PPO's performance, we conduct a grid search for both environments using the default reward function.

In order to model human feedback, we make use of a synthetic oracle that has access to the true reward function. The preferences of the synthetic oracle are according to the ones outlined in Sec. III-A. An oracle prefers a segment over the other if the cumulative reward is higher. To emulate the input of a human teacher, we incorporate noise into the system by deliberately mislabelling 10% of the preferences provided. Both policy π and $\hat{\mathcal{R}}$ are asynchronously updated. Policy π is updated every 1024 steps. To update the reward function $\hat{\mathcal{R}}$, query samples are obtained through the different query selection methods and filled by a common synthetic oracle. Then, we use the batch of queries to optimize $\hat{\mathcal{R}}$ every 20K steps, according to eq.2. In each $\hat{\mathcal{R}}$ update we request 1/10 of the total planned queries. For all environments the length L of the segments making up the queries are set to 50. We employ neural networks of different architectures. The reward network $\hat{\mathcal{R}}$ for both environments has a hidden layer size of (256, 256, 256) with ReLu activations. The policy π and value networks for the Cheetah environment consist of hidden layers of size (256, 256) with ReLu activations, while the Walker2D environment uses hidden layers of size (64, 64) with Tanh activations respectively.

Additionally, we utilize a VAE to acquire a latent space for VARIQuery. For Walker and Cheetah, the encoder contains a hidden layer size of (128, 64, 32), whereas for the social navigation environment, it contains a hidden layer size of (256, 128, 64). Similarly, the decoder comprises the same sizes in reverse order. Consequently, the latent vectors we obtain are of size 64 for the social navigation environment and 32 for Cheetah and Walker.

B. Exploring the impact of query selection methods

To answer Q1, we implement all query selection strategies presented in Sec. III-B. We re-implement into our own framework the entropy-based sampling approach presented by Lee et al. [39] which the authors made available as an open-source implementation. The ensemble-based sampling can be considered an ablation of VARIQuery as we also order queries by the variance of the ensemble as part of our approach (see Sec. IV). We assess the performance of each query selection method using 400 and 800 queries. In

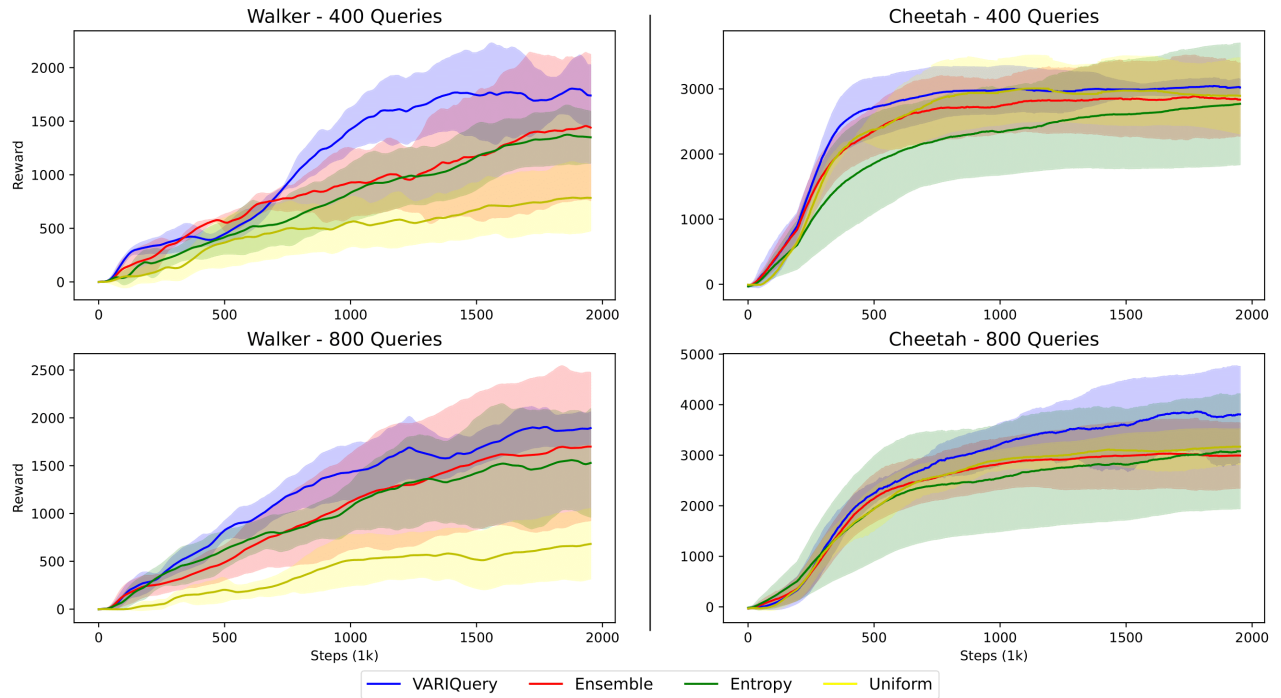


Fig. 3: The figure presents the learning curves obtained by four different sampling methods: VARIQuery, ensemble-based, particle-based entropy-based, and uniform sampling, for both Walker2D and Cheetah environments. The solid lines represent the mean, and the shaded regions represent the standard deviation. All conditions were run for both 400 and 800 queries over a total of 1×10^6 environment steps.

Fig. 3 are the complete outcomes of executing VARIQuery and alternative query selection methods.

Walker2D. In Walker2D we observe a gradient in performance for the considered query selection methods. By order of highest to lowest performance: VARIQuery, ensemble-based, entropy-based, and uniform sampling. We find uniform sampling to perform significantly worse regardless of the number of queries considered. Both VARIQuery and ensemble-based sampling perform similarly in this environment, with VARIQuery only outperforming ensemble-based sampling by $\sim 11\%$. Nevertheless, our results show that both methods present better learning performance and sample complexity by outperforming uniform and entropy-based sampling. Indeed, VARIQuery presents a similar learning performance to entropy-based sampling for half the queries.

Cheetah. The experiments for Cheetah reveal more expressive outcomes (see Fig. 3). Both ensemble-based, entropy-based and uniform sampling methods, appear to plateau around $\sim 3K$ environmental reward, for both 400 and 800 queries. These findings are consistent with previous research [5], [6]. Although VARIQuery shows comparable performance with 400 queries, we observe a significant improvement in environmental reward performance for 800 queries. This improvement amounts to almost $\sim 21\%$ compared to other query selection methods.

C. Underlying query structure

In this section, we utilize VARIQuery to examine the latent space representations of selected queries in both the Cheetah and Walker environment. We first train an agent using VARIQuery for 100K timesteps, and then we collect a sample of 40 queries to demonstrate their distribution. Using

t-distributed stochastic neighbor embedding (t-SNE), we project the latent representation of the queries and sampled segments into 2D space to visualize the underlying structure. Fig.4 highlights how the latent space creates clusters of similar segments, with clearly distinguished k -NN clusters. By sampling segments based on clusters, we ensure that queries represent all clusters in the latent space. Additionally, we ensure the segments making up our queries are sampled from different k -NN clusters (see Sec. IV-C), which are visually represented by the lines drawn between different segments forming a single query, showing support for Q2.

VI. VARIQUERY WITH HUMAN FEEDBACK

A. Social Navigation

To test VARIQuery with human feedback we use an additional social navigation environment introduced in [46]. The environment contemplates a robot, walls, navigation goals and three moving humans in a narrow corridor (see Fig.5). The robot observes the environment through simulated lidar rays which are one-hot encoded with object information and distance. Since the environment is quite challenging due to its high dimensionality, we bootstrap from a performant policy to save human feedback from teaching a robot to avoid collisions.

B. Procedure

To better understand Q3, we collect feedback in an online human study. We implement a browser-based framework on Amazon Mechanical Turk (AMT) which presents pairs of videos of trajectory segments from a policy. Each video has the same duration of 4 seconds since queries are evaluated on segments of equal length. To keep the initial conditions

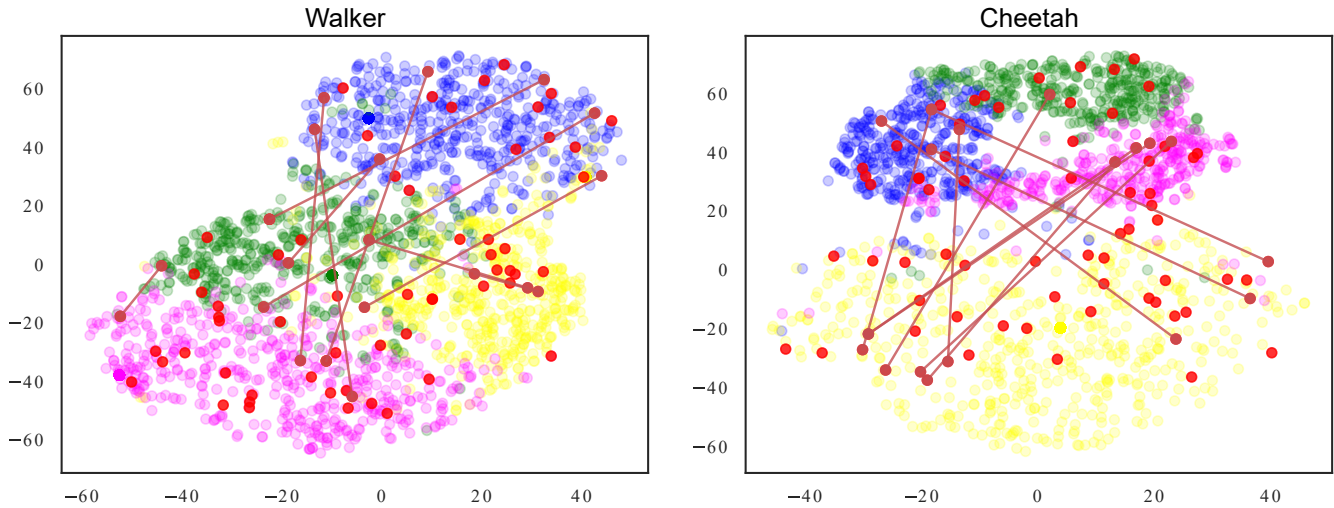


Fig. 4: Using t-SNE in the Walker and Cheetah environments, we show the 2D projection of the latent representation learned by the VAE. The various colors represents state-sequences assigned to different k -NN clusters on the latent space. Red points are segments selected for queries and the red lines show example distances between segments forming a query pair.

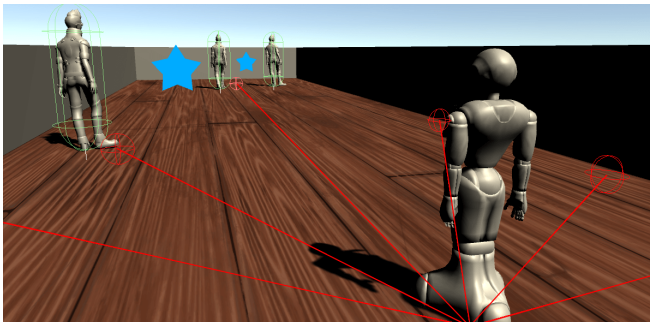


Fig. 5: The social navigation environment. The robot has to navigate to the end goal while avoiding humans walking in the environment. The robot can observe the environment using a lidar.

equal, we start the preference learning process with the same policy π , and reward model $\hat{\mathcal{R}}$ parameters and a dataset of unlabelled segments \mathcal{D}_{Un} . We quantify human effort as time spent per query answered and the number of times a human has an impasse when giving a preference, resulting in a skip, which we consider an indirect measure of query redundancy.

C. Participants

We recruited a total of 70 crowdworkers, with 24 using VARIQuery, 23 using entropy-based sampling, and 23 using uniform sampling. Of the participants, 40% were female and 60% were male, with ages ranging from 20 to 70 years. Ten participants were excluded for failing an attention check. All participants were from the United States.

D. Human effort results

We found that users expressed no preference for 64 pairs using VARIQuery and 88 pairs using uniform sampling. To compare the differences between VARIQuery and uniform sampling, we used Pearson’s χ^2 -test for independence with a 2x2 contingency table. The test yielded a result of ($\chi^2 = 4.3, p < 0.05$), showing statistical significance, indicating that users were able to express a preference more often using VARIQuery than with uniform sampling. For entropy

sampling, users expressed no preference for 75 pairs. While we observed a trend of more users expressing no preferences with entropy sampling, the difference to VARIQuery did not reach statistical significance ($\chi^2 = 0.87, p = 0.35$).

Upon measuring the time participants took to provide preferences over all queries, we observed that the average time per query was 21.2 ± 9.5 seconds for VARIQuery; 23 ± 9.35 seconds for entropy-based sampling; 24.35 ± 8.9 seconds for uniform sampling. This indicates that VARIQuery required approximately 13% less time than uniform sampling and 8.5% less time than entropy sampling. Therefore, it is possible that users might find it easier to provide preferences with sampling methods that encourage dissimilarity within queries. Videos of different policies can be found on the supplemental materials of this paper.

VII. CONCLUSIONS

In this paper, we present VARIQuery, a novel query selection algorithm for preference-based RL. Our experimental results show a clear trend with VARIQuery consistently performing on-par or above other state-of-the-art methods. VARIQuery achieves this by clustering the latent representation of trajectory segments. To better understand the dissimilarity of segments within queries, we offered additional insight in the form of t-SNE visualizations. We conducted a user study collecting actual human preferences elicited with query batches produced by different query selection methods. Our findings indicate our approach was able to produce queries which take less time to evaluate and are less prone to cause an impasse in $\sim 27\%$ and $\sim 13\%$ for uniform and entropy-based sampling respectively.

REFERENCES

- [1] J. MacGlashan, M. K. Ho, R. Loftin, B. Peng, G. Wang, D. L. Roberts, M. E. Taylor, and M. L. Littman, “Interactive learning from policy-dependent human feedback,” in *Int. Conf. on Machine Learning*. PMLR, 2017, pp. 2285–2294.

- [2] S. H. Huang, I. Huang, R. Pandya, and A. D. Dragan, "Nonverbal robot feedback for human teachers," *arXiv preprint arXiv:1911.02320*, 2019.
- [3] R. Akrou, M. Schoenauer, and M. Sebag, "April: Active preference learning-based reinforcement learning," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II* 23. Springer, 2012, pp. 116–131.
- [4] C. Wirth, R. Akrou, G. Neumann, J. Fürnkranz *et al.*, "A survey of preference-based reinforcement learning methods," *Journal of Machine Learning Research*, vol. 18, no. 136, pp. 1–46, 2017.
- [5] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, 2017.
- [6] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, "Reward learning from human preferences and demonstrations in atari," *Advances in neural information processing systems*, vol. 31, 2018.
- [7] S. Dasgupta, "Two faces of active learning," *Theoretical computer science*, vol. 412, no. 19, pp. 1767–1781, 2011.
- [8] D. P. Kingma, M. Welling *et al.*, "An introduction to variational autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019.
- [9] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A survey of deep active learning," *ACM computing surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021.
- [10] J. Lin, L. Zhao, S. Li, R. Ward, and Z. J. Wang, "Active-learning-incorporated deep transfer learning for hyperspectral image classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 11, pp. 4048–4062, 2018.
- [11] A. J. Joshi, F. Porikli, and N. Papanikolopoulos, "Multi-class active learning for image classification," in *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 2372–2379.
- [12] H. S. Seung, M. Oppor, and H. Sompolinsky, "Query by committee," in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 287–294.
- [13] B. Settles, M. Craven, and S. Ray, "Multiple-instance active learning," *Advances in neural information processing systems*, vol. 20, 2007.
- [14] N. Roy and A. McCallum, "Toward optimal active learning through monte carlo estimation of error reduction," *ICML, Williamstown*, vol. 2, pp. 441–448, 2001.
- [15] A. Freytag, E. Rodner, and J. Denzler, "Selecting influential examples: Active learning with expected model output changes," in *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part IV* 13. Springer, 2014, pp. 562–577.
- [16] M. Bilgic and L. Getoor, "Link-based active learning," in *NIPS Workshop on Analyzing Networks and Learning with Graphs*, vol. 4, 2009, p. 9.
- [17] Y. Guo, "Active instance sampling via matrix partition," *Advances in Neural Information Processing Systems*, vol. 23, 2010.
- [18] H. T. Nguyen and A. Smeulders, "Active learning using pre-clustering," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 79.
- [19] C. Shui, F. Zhou, C. Gagné, and B. Wang, "Deep active learning: Unified and principled method for query and training," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 1308–1318.
- [20] C. Yin, B. Qian, S. Cao, X. Li, J. Wei, Q. Zheng, and I. Davidson, "Deep similarity-based batch mode active learning with exploration-exploitation," in *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2017, pp. 575–584.
- [21] F. Zhdanov, "Diverse mini-batch active learning," *arXiv preprint arXiv:1901.05954*, 2019.
- [22] T. He, X. Jin, G. Ding, L. Yi, and C. Yan, "Towards better uncertainty sampling: Active learning with multiple views for deep convolutional neural network," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2019, pp. 1360–1365.
- [23] H. Ranganathan, H. Venkateswara, S. Chakraborty, and S. Panchanathan, "Deep active learning for image classification," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3934–3938.
- [24] M. Ducoffe and F. Precioso, "Adversarial active learning for deep networks: a margin based approach," *arXiv preprint arXiv:1802.09841*, 2018.
- [25] S. Sinha, S. Ebrahimi, and T. Darrell, "Variational adversarial active learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5972–5981.
- [26] K. Kim, D. Park, K. I. Kim, and S. Y. Chun, "Task-aware variational adversarial active learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8166–8175.
- [27] D. Yoo and I. S. Kweon, "Learning loss for active learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 93–102.
- [28] Y. Saquil, K. I. Kim, and P. M. Hall, "Ranking cgans: Subjective control over semantic image attributes," in *British Machine Vision Conference*, 2018.
- [29] J. Hejna and D. Sadigh, "Few-shot preference learning for human-in-the-loop rl," *arXiv preprint arXiv:2212.03363*, 2022.
- [30] A. Wilson, A. Fern, and P. Tadepalli, "A bayesian approach for policy learning from trajectory preference queries," *Advances in neural information processing systems*, vol. 25, 2012.
- [31] R. Akrou, M. Schoenauer, and M. Sebag, "Preference-based policy learning," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5–9, 2011. Proceedings, Part I* 11. Springer, 2011, pp. 12–27.
- [32] R. Akrou, M. Schoenauer, M. Sebag, and J.-C. Souplet, "Programming by feedback," in *International Conference on Machine Learning*, no. 32. JMLR. org, 2014, pp. 1503–1511.
- [33] J. Fürnkranz, E. Hüllermeier, W. Cheng, and S.-H. Park, "Preference-based reinforcement learning: a formal framework and a policy iteration algorithm," *Machine learning*, vol. 89, pp. 123–156, 2012.
- [34] A. Jain, S. Sharma, T. Joachims, and A. Saxena, "Learning preferences for manipulation tasks from online coactive feedback," *The International Journal of Robotics Research*, vol. 34, no. 10, pp. 1296–1313, 2015.
- [35] R. Holladay, S. Javdani, A. Dragan, and S. Srinivasa, "Active comparison based learning incorporating user uncertainty and noise," in *RSS Workshop on Model Learning for Human-Robot Communication*, 2016.
- [36] D. Sadigh, A. D. Dragan, S. Sastry, and S. A. Seshia, "Active preference-based learning of reward functions," in *Robotics: Science and Systems*, 2017.
- [37] E. Bıyık, N. Huynh, M. J. Kochenderfer, and D. Sadigh, "Active preference-based gaussian process regression for reward learning," in *Robotics: Science and Systems (RSS)*, 2020.
- [38] E. Bıyık, D. P. Losey, M. Palan, N. C. Landolfi, G. Shevchuk, and D. Sadigh, "Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences," *The International Journal of Robotics Research*, vol. 41, no. 1, pp. 45–67, 2022.
- [39] K. Lee, L. Smith, and P. Abbeel, "Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training," *arXiv preprint arXiv:2106.05091*, 2021.
- [40] H. Liu and P. Abbeel, "Behavior from the void: Unsupervised active pre-training," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 459–18 473, 2021.
- [41] T. Scheffer, C. Decomain, and S. Wrobel, "Active hidden markov models for information extraction," in *Advances in Intelligent Data Analysis: 4th International Conference, IDA 2001 Cascais, Portugal, September 13–15, 2001 Proceedings* 4. Springer, 2001, pp. 309–318.
- [42] P. Viappiani and C. Boutilier, "Optimal bayesian recommendation sets and myopically optimal choice query sets," *Advances in neural information processing systems*, vol. 23, 2010.
- [43] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" in *Database Theory—ICDT'99: 7th International Conference Jerusalem, Israel, January 10–12, 1999 Proceedings* 7. Springer, 1999, pp. 217–235.
- [44] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.
- [45] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
- [46] D. Marta, C. Pek, G. I. Melsión, J. Tumova, and I. Leite, "Human-feedback shield synthesis for perceived safety in deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 406–413, 2021.