RoBERT: Low-Cost Bi-Directional Sequence Model for Flexible Robot Behavior Control

Anonymous authors
Paper under double-blind review

Abstract

Requirement of human involvement for data collection or system design has always been a major challenge for building robot control policy. In this paper, we present **Ro**bot-**BERT** (RoBERT), a method to build general robot control policy for complex behaviors with *least* human effort. Starting from unsupervisedly-collected dataset, RoBERT has no requirements of human labels, high-quality behavior dataset or accurate information of system model, in contrast to most other methods for building general robot agent. RoBERT is further unsupervisedly trained via *Masked Action-Inverse-Inference* (MAII), a method inspired by *Masked Language Modeling* (MLM) in BERT-like language models and has potential to enable *zero-shot*, *multi-task*, *keyframe-based* robot control with little architectural change and user-friendly interface. In our empirical study, RoBERT is successfully applied on various types of robots in simulated environment and could generate stable and flexible behaviors to fulfill complex commands.

1 Introduction

Recent years have witnessed a surge of research interest to apply Transformer-based (Vaswani et al., 2017) large sequence model (a.k.a Large Language Model, LLM) into multiple machine learning related fields, such as Computer Vision (CV) (Dosovitskiy et al., 2021), Natural Language Processing (NLP) (Devlin et al., 2019; He et al., 2020; Brown et al., 2020) and Decision Making (Reed et al., 2022; Lee et al., 2022; Chen et al., 2021; Janner et al., 2021).

Among them, using pretrained large sequence model, combined with algorithm in Reinforcement Learning (RL) or Imitation Learning (IL) to build general robotic agent is of much interest and importance (Brohan et al., 2023b;a; Huang et al., 2023), as conventional model-based method usually struggles in complex robotic system (Lyu & Cheah, 2020; Cui et al., 2021).

However, existing works on this field either rely on large amount of human-labelled multi-modal data (Brohan et al., 2023b;a; Huang et al., 2023) or near-optimal behavior dataset (Reed et al., 2022; Brohan et al., 2023a) that are expensive to collect. In our insight, the need of such human effort would hinder the development of large sequence model for robot control, owing to the lack of reusable data and expert resources in community of Robotics.

Inspired by recent advancements in *Reward-free* and *Unsupervised RL* (Jin et al., 2020; Tarbouriech et al., 2020; Chen et al., 2022; Yarats et al., 2021; Liu & Abbeel, 2021b; Laskin et al., 2021; Burda et al., 2018b), we propose a new way to both collect dataset and train models with little human effort. It means: 1) Dataset is collected by maximizing *intrinsic rewards* without any *extrinsic reward* (and *reward engineering*). 2) Models are then trained directly and purely on this collected dataset following methods inspired by Behavioral Cloning (BC) and Curriculum Learning (CL) (Wang et al., 2022). 3) During deployment, user-supplied commands *extract* behaviors needed, enabling *goals-conditioned* or even *keyframe-based open-loop* control *zero-shotly*.

What's more, a method trained in this way is naturally *multi-task*. Our insight is that every specified reward function would introduce a *bias* towards what behaviors an agent should generate. By encouraging an agent to perform task in one direction, we are discouraging it from performing task in opposite direction. For example, by tweaking reward function to train a robot to teach dance smoothly, we're restricting its ability to serve as a counter-example for showing how un-smooth dances look like.

In summary, the main contributions of this paper are as follows:

1. We present RoBERT, a method to both collect dataset and train sequence models at low cost. To the best of our knowledge, our method requires *the least* amount of human labelling, expert behavior dataset or system knowledge, compared to other state-of-the-art methods for general robot control/decision-making.
2. To the best of our knowledge, we are the first to show that with recent advancements in unsupervise RL and computing ability, RoBERT could be readily instantiated on rather complex robot models (Universal Robot 5e robot arm and Shadow E3M5 robot hand) with dataset of only millions of transitions.
3. We perform extensive experiments and analysis to demonstrate our instantiated RoBERT could successfully generate flexible and stable actions on various robot models and are robust against hyperparameters and model architecture. We also plan to release dataset and code used in this work as a testbed for future researches.

2 Related Work

2.1 Sequence Modeling in Decision Making

Applying sequence model into field of decision making has been a popular research direction since the recent success of LLM in NLP area. Chen et al. (2021) and Janner et al. (2021) firstly introduce Transformer into model-free and model-based RL method respectively and show promising results in many RL tasks. Following their work, Furuta et al. (2021) further investigates various ways to integrate Transformer into various RL/IL setting and Kim et al. (2023) integrates Transformer into Preference-based RL (PbRL). In IL, Shafiullah et al. (2022) proposes to use Transformer for multi-modal behavior modelling and their idea is evaluated into our method. Besides, some works also focus on Representation Learning by masking on bi-directional Transformer encoder-decoder, as in Liu et al. (2022); Wu et al. (2023). Our work mainly differs from theirs as ours focuses on instantiating a control policy from dataset collected purely from intrinsic rewards and show that decent performance coule be achieved with only millions of transitions.

2.2 Reward-Free and Unsupervise RL

Reward-free RL proposes to study how to perform exploration without knowing task/reward function *a prior* and collects dataset which could enable near-optimal policy learning for *any* task specified afterwards (Jin et al., 2020; Tarbouriech et al., 2020; Chen et al., 2022). Many works in this direction focus on theoretical results. More practically, Unsupervise RL employs various *intrinsic reward* (prediction error, distribution entropy, etc.) to maximize statistical properties of trajectories collected, in order to collect diverse behavior dataset or learn task policy efficiently. Burda et al. (2018a); Laskin et al. (2021); Yarats et al. (2022); Liu & Abbeel (2021b); Lobel et al. (2023); Liu & Abbeel (2021a); Yarats et al. (2021); Eysenbach et al. (2018) are all works in this line that differ mainly on optimization objectives selected. In this work, we employ various unsupervise RL methods and mix datasets their collected, as a resource of diverse behaviors (more in Section 3.2).

2.3 Robotics

Applying Neural Network (NN) and RL/IL into field of robot control has gained great achievements recently. For example, on locomotion task, quadruped robot has been successfully trained to move on various terrains with ability of fast-adaptation or self-recovery (Nahrendra et al., 2023; Choi et al., 2023; Miki et al., 2022; Yang et al., 2020). In simulation environment, Peng et al. (2018); Tessler et al. (2023); Peng et al. (2022); Chen et al. (2023) are also works commanding robot character performing complex sequential behaviors for animation generation. Recently, works combining LLMs and Robotics have also emerged. In Brohan et al. (2023b;a), VLA (Vision-Language-Action) Model based control policy could enable robot arm to follow human instruction and complete various tasks. Also, Huang et al. (2023) tries to employ LLM as high-level planner and is also evaluated on robot arm. Different from theirs, ours investigates how to use Transformer model for Robotics in a low-cost and accessible way.

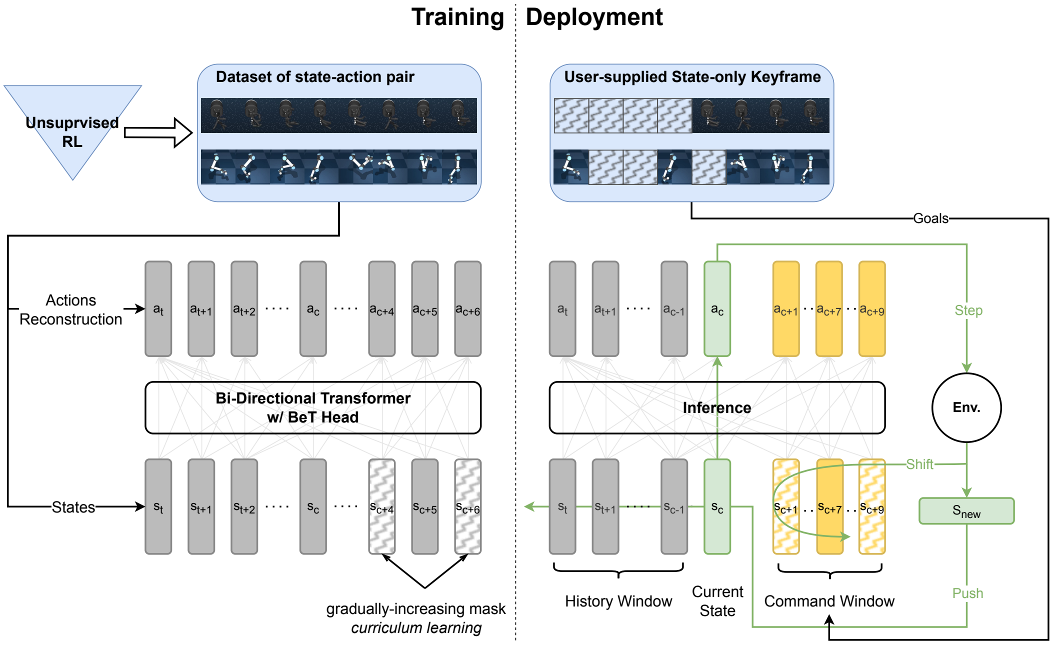

Training
Deployment
Unsuprvised RL
Dataset of state-action pair
User-supplied State-only Keyframe
Goals
Actions Reconstruction
a_t a_t+1 a_t+2 a_c a_c+4 a_c+5 a_c+6
Bi-Directional Transformer w/ BeT Head
States
s_t s_t+1 s_t+2 s_c s_c+4 s_c+5 s_c+6
gradually-increasing mask
curriculum learning
a_t a_t+1 a_c-1 a_c a_c+1 a_c+7 a_c+9
Step
Inference
Env.
Shift
s_t s_t+1 s_c-1 s_c s_c+1 s_c+7 s_c+9
S_new
Push
History Window
Current State
Command Window

Figure 1: Framework of RoBERT. States with zigzagged line represents masked ones.

3 RoBERT

The framework of RoBERT is given in Figure 1. In the training phase, trajectories of state-action pair are feed into Bi-Directional Transformer with gradually-increasing (CL-based) masking applied in the same 'command window' of deployment. During deployment, the model is queried with input concatenated from 'history window', current state and 'command window', then the output action corresponding to current state is rollouted in environment. After each rollout, new state would be back-pushed into history-current window while command window would be left-shifted.

3.1 Preliminiaries

Consider a rewardless Markov Decision Process (MDP), $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{S}_0)$ where $\mathcal{S}$ is set of all states, $\mathcal{A}$ is set of all actions, $\mathcal{T}$ is transition dynamics $\mathcal{T}(\mathcal{S}'|\mathcal{S}, \mathcal{A}) \to \mathcal{R}$ that defines probability distribution of next states given previous state and action taken, and $\mathcal{S}_0$ is distribution of initial states. Objective of Imitation Learning is to find a policy π that imitates behaviors present in a dataset $D = \{(s, a)\}$:

$$\pi = \arg\min_{\pi} \mathop{\mathbb{E}}_{s,a\sim D} \mathcal{L}(\pi(s), a)$$

where $\mathcal{L}$ is some loss function (e.g. Mean Square Loss (MSE) when π is deterministic or Softmax between loglikelihood when π is stochastic and action set is finite). Finally, we introduce *inverse dynamics function* $\mathcal{I} = \mathcal{I}(s_t, ..., s_{t+h}) \to (a_t, ..., a_{t+h})$ that outputs action sequence corresponding to input state sequence.

3.2 Traversal of State Sequence Space

Consider a dataset of trajectory of length h, $D = \{(s_t, a_t, s_{t+1}, a_{t+1}, ..., s_{t+h}, a_{t+h})\}_{t=1}^{N}$, note s_t isn't necessarily initial state s_0 and could be any state of a trajectory, so long as length of its following steps exceeding h. If we obtain such a dataset that distribution of its state sequence ($D_s = \{(s_t, s_{t+1}, ..., s_{t+h})\}_{t=1}^{N}$) is diverse enough, i.e.:

$$\forall s_q^h = (s_q, s_{q+1}, ..., s_{q+h}) \in \{\text{all possible state seq.}\} \implies$$

$$\exists s_i^h = (s_i, s_{i+1}, ..., s_{i+h}) \in D_s \quad s.t. \quad \mathcal{L}(s_i^h, s_q^h) \le \theta$$

where $\mathcal{L}$ is some loss function, $\theta \in \mathcal{R}^+$ is a measure of diversity of D_s and while θ approaching 0, D_s is more diverse in state sequence space.

Then, if we have a near-optimal policy π:

$$\pi \quad s.t. \quad \forall s_i^h \in D_s,\ \mathcal{L}(a_i^h, \pi(s_i^h)) \leq \epsilon$$

where $\mathcal{L}$ is loss function, $a_i^h = (a_i, a_{i+1}, ..., a_{i+h})$ is series of actions corresponding to s_i^h, $\pi(s_i^h) = (\pi(s_i), \pi(s_{i+1}), ..., \pi(s_{i+h}))$ is action sequence output by π, $\epsilon \in \mathcal{R}^+$ is a measure of optimality of π and while ϵ approaching 0, π is more optimal.

The insight here is the data-driven learning process above could yield a decent function approximator for inverse dynamics, i.e.:

$$\text{for } \pi = \arg\min_{\pi} \mathop{\mathbb{E}}_{s^h \in D_s} [\mathcal{L}(a^h, \pi(s^h))] \ \text{ and } \ \forall s_q^h \in \{\text{all possible state seq.}\}$$
$$\text{we have } \ \mathcal{L}(\mathcal{I}(s_q^h), \pi(s_q^h)) \leq \delta$$

where $\mathcal{L}$ is loss function and $\delta \in R^+$ is a bounded value.

Therefore, with a near-optimal π on a decently diverse dataset D, we could perform multi-goals conditioned control by concatenating history window, current state and remaining goals to query π for recovered action series.

3.3 Masked Action-Inverse-Inference (MAII)

Despite π's ability to serve as a multi-goals conditioned policy, specifying series of states as goals is still a tedious process and not user-friendly. Therefore, inspired by Masked Language Modeling (MLM) in BERT (Devlin et al., 2019) and Curriculum Learning (CL), we design MAII, a way to introduce masking into training of RoBERT for keyframes control. In details, user-supplied commands could contain mask token same as the one applied during training phase, indicating 'no goal specified' on these keyframes.

An intuitive idea here is as masks increasing, determinism of inverse dynamics would decrease and our setting would become a multi-modality Imitation Learning, as studied in Shafiullah et al. (2022). Hence, we also test technique in Shafiullah et al. (2022): output index of kmeans cluster and offset to it, then combine them back to continuous actions (please refer to Shafiullah et al. (2022) for details).

3.4 Keyframe-Based Open-Loop Control

After training, model is fixed and tested as a zero-shot policy. As Figure 1 shows, model keeps a history window that tracks last few states and a future/command window specifying remaining commands to achieve. With insertion of current state in between, this composes a state sequence same as training time.

We choose only position values of joints as states in this work because 1) they are easy to interpret, specify and tweak. 2) a well-trained model shall have ability to infer velocity and other higher-order derivatives automatically given a series of position observations. We also choose open-loop control since close-loop control conflicts with time-aware keyframe control.

4 Main Experiments

4.1 General Setting

We briefly describe general setting of our main experiments in this section. Then we present our main results on 3 distinct robots: Pointmass, Universal Robot 5E arm (UR5E) and Shadow Hand robot hand (Shadow Hand) as shown in Figure 2. For more details of our experiment settings, please refer to Appendix A.1 and our code.

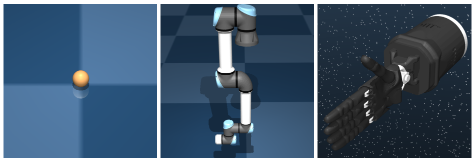


Figure 2: Environments used: (from left to right) Pointmass, UR5E and Shadow Hand

4.1.1 Dataset Collection

As explained in Section 2.2, we employ existing unsupervise RL methods adopted from URLB (Laskin et al., 2021) codebase and combine dataset collected by them . We choose dataset size as a *rule of thumb* and tend to collect excessive amount as shrinking dataset is cheaper than re-collecting.

Table 1: Outline of environment, dataset and model settings

Env.	**State Dim.**	**Action Dim.**	**#. Transitions[a] Collected**	**#. Model Params.**
Pointmass	2	2	306M[b]	0.11M ∼ 50.4M
UR5E	5	5	41.9M	42.6M
Shadow Hand	22	18	131.3M	42.73M

[a] Transitions here refer to one step of state-action.
[b] We deliberately collect excessive amount of data for testing scaling effects, as shown in Section 4.2.1.

4.1.2 Training

During training, we firstly split collected trajectory dataset into training and test set (95%-5% ratio). From test set, we randomly extract dozens of trajectories and combine them with several manual-designed tasks (for instance, *stay in the origin*) to form the final test set. Then we train our model on training set only (default hyperparameters in Appendix A.2) and calculate *Precision* (Mean Absolute Error (MAE)) between positions achieved and positions desired[1] as a metric of performance and pick best model based on it. Also note we calculate precision over strictly aligned timeframes and don't allow time-warping.

4.1.3 Evaluation - Task Fulfillment

We design experiments of task fulfillment to test how well RoBERT could generate behaviors to fulfill given commands/tasks in a keyframe-based manner. All experiments are conducted in a single episode without reset. All tasks span over 10 seconds while some exceeding 20 seconds. It's also worth noting that all tasks are *unknown* during data collection and model training phase, hence illustrating general and multi-task ability of RoBERT.

For each robot, we present visual results (2D coordinates plotting for pointmass or video clips for others) and diagram of *Precision-Time* to illustrate time-axis stability of RoBERT. We highly recommend readers to refer to videos in supplement material (Appendix A.4 and uploaded file) for a best view.

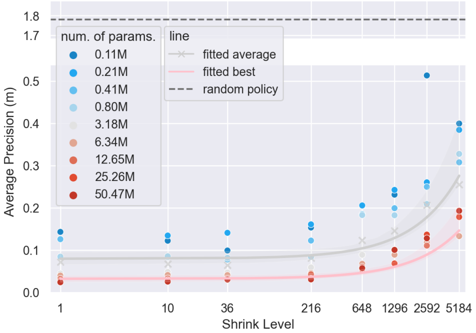


Figure 3: Scaling effects of Pointmass: *shrink* means by which ratio the original dataset is reduced and *Average Precision* is average of lowest precision observed in their training process. Lightgrey line is fitted for average values in a shrink while pink line is fitted for best value. Translucent bands are 95% confidence interval.

4.2 Pointmass

A pointmass is controlled by 2 slide joints on an infinitely large plane. State space is 2 cartesian coordinates (xy) of it. We perform direct force control in this environment.

4.2.1 Scaling Effects

We firstly investigate relationship between size of collected dataset and performance of trained model in order to guide our following data collection and training on more complex environments (Details in Appendix A.1.1).

As observed in Figure 3: 1) As dataset size reduces, models with various size and learning rate obtain worse performance. 2) In pointmass environment, effects of saturation is observed as enlarging dataset

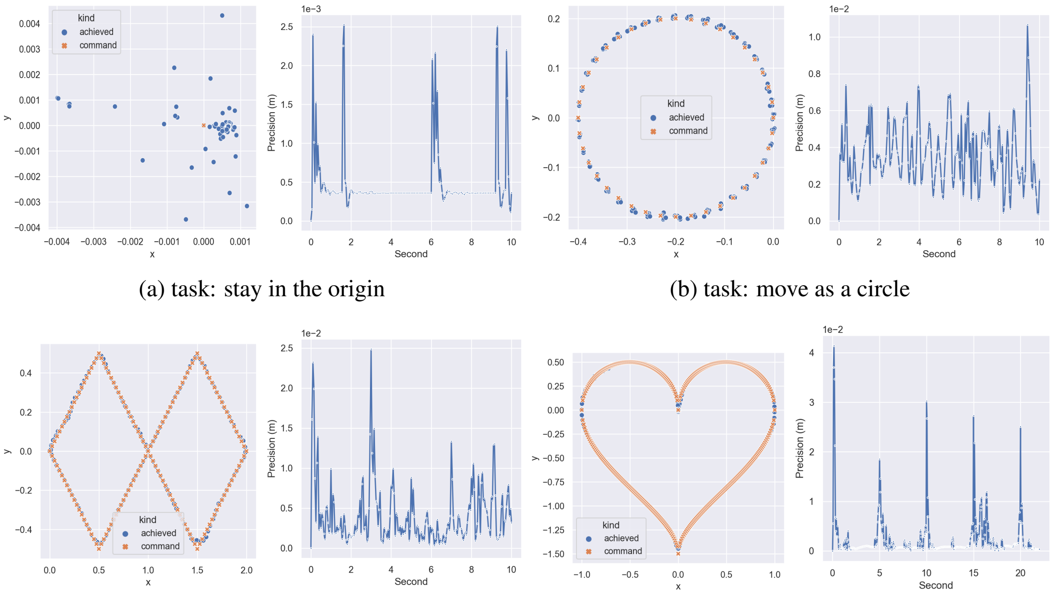
[1] Masked keyframes are excluded from precision calculation but included in loss calculation.

brings only insignificant performance increase. 3) Larger model usually obtains better results, compared to small-sized one.

Because we use precision as the metric, a relatively small difference in precision may already cause a severe performance degradation. To illustrate this, we choose the best model in shrink 5184 and test its performance on task fulfillment as in Section 4.2.2. Recorded videos are given in supplement materials and also aligns with our expectation that performance suffers from significant degradation.

4.2.2 Performance on Task Fulfillment

In this section, we instruct RoBERT to fulfill multiple complex tasks. We choose the best model in shrink 10 for task fulfillments and for each task, we draw plots of trajectory on 2D coordinate and their tracking loss. We also note that commands given here may itself be *physically impossible* such as the sharp turn in *triangle* task. Results are shown in Figure 4.



(a) task: stay in the origin

(b) task: move as a circle

(c) task: move as a triangle

(d) task: move as a heart curve

Figure 4: Results of RoBERT performing multi-task in pointmass (best viewed in videos). In coordinate plot, circle point is achieved position while crossmark point is commanded point.

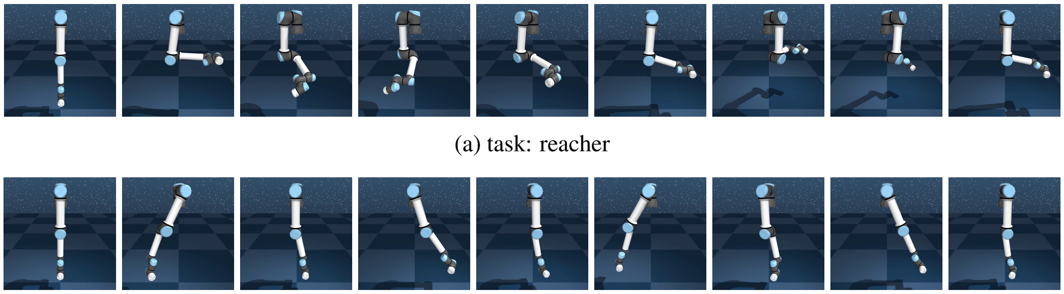
4.3 Robot Arm - UR5E

An UR5E robot arm is controlled by 5 hinge joints with its base fixed in the air and normal gravity enabled. State space is 5 radian positions of each joint. Action space is torque directly applied to each joint.

As illustrated in Figure 3, models with different size and hyperparameters tend to have similar performance when dataset is decently large and diverse. Thus we don't perform hyperparameter-tuning on UR5E and ShadowHand. This also aligns with our insight that with recent progress in unsupervise RL we could already obtain a general robot control policy readily.

4.3.1 Performance on Task Fulfillment

In this section, we present results of commanding RoBERT-UR5E to perform multiple tasks. The first task is to perform a 'reacher'-like task where robot arm needs to rotate while keeping *elbow joint* fixed to 90 degrees. Second one is to 'swing' like a clock tick while moving only *shoulder lift joint*. Videos clips are shown in Figure 5.



(a) task: reacher

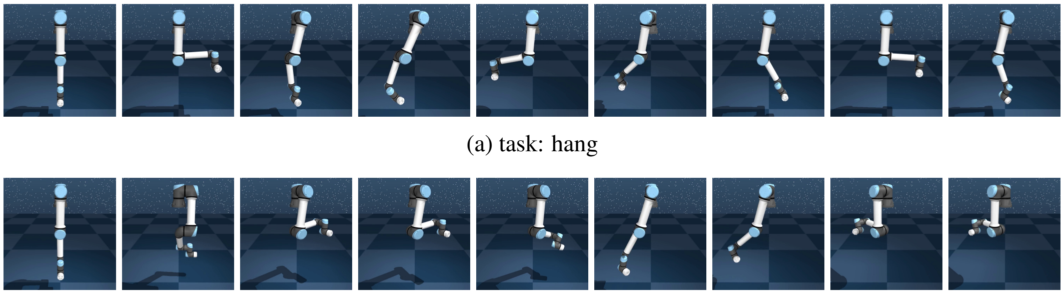
(b) task: swing

Figure 5: Video clips of RoBERT controlling UR5E (best viewed in videos).

4.3.2 Performance on Keyframe-based Control

In this section, we test RoBERT-UR5E's ability to fulfill commands when masks enabled.

The first command is to let robot arm 'hang' its elbow joint in left and right side alternatively, while keeping other parts fixed. All transition frames that happen in between of these 2 fixed posture are masked. The second command is commanding robot arm to reach front-left and back-left (named 'front_back') alternatively while masking transition frames.



(a) task: hang

(b) task: front_back

Figure 6: Video clips of RoBERT controlling UR5E (best viewed in videos).

4.4 Robot Hand - Shadow Hand

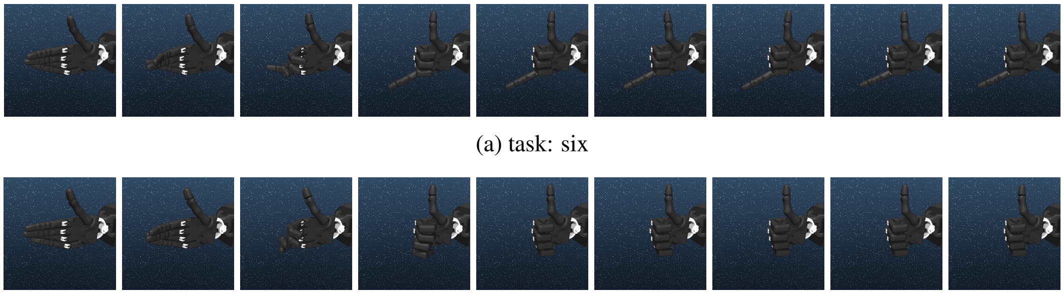
A Shadow Hand with 22 hinge joints are controlled by 18 actuators. There are 14 actuators applying torques directly on 14 hinge joints while 4 actuator applying control signal on 4 tendons, each linking 2 joints. State space is radian positions of 22 joints. Action space is control signal applied in each actuator. Gravity is removed in this environment.

4.4.1 Performance on Keyframe-based Control

We test how well RoBERT could control Shadow Hand performing various tasks in *keyframe-based* manner. 2 tasks in Figure 7 are 1) command hand to pose a 'six' posture. 2) command hand to pose a 'thumb' posture, respectively. In these tasks, transition frames that leading hand from initial posture to desired ones are masked.

4.4.2 Performance on Sequential Keyframe-based Control

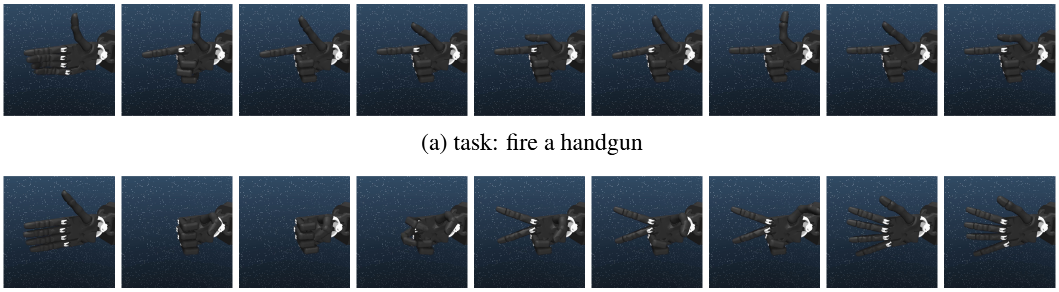
Due to the nature of RoBERT, it's easy to command for sequential and dynamic behaviors. In this section, we demonstrate 2 commands that test this feature of RoBERT. The first one is to perform 'fire a handgun' while the second one is playing 'rock-scissor-paper' game, as shown in Figure 8. In these tasks, transition frames leading hand from initial posture to desired ones are masked.



(a) task: six

(b) task: thumb

Figure 7: Video clips of RoBERT controlling ShadowHand (best viewed in videos).



(a) task: fire a handgun

(b) task: rock-scissor-paper

Figure 8: Video clips of complex behaviors of RoBERT on Shadow Hand (best viewed in video).

4.5 Analysis

In this section, we investigate how several aforementioned components affect model performance: 1) How robust is RoBERT against ill-collected dataset? 2) How does number of action bins affect model performance? 3) Is CL-based masking improving model performance?

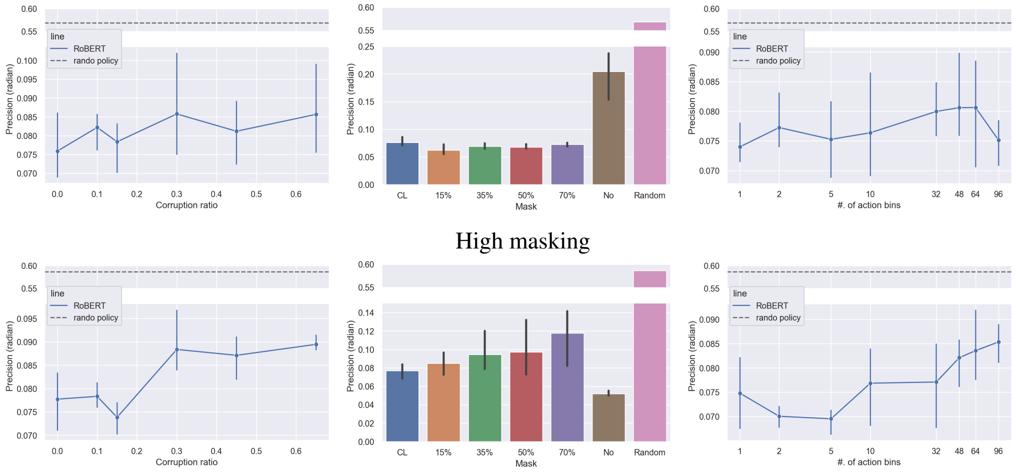
To answer these questions, we choose ShadowHand and evaluate its performance over 2 sets of tasks extracted from test set in training process: 1) 'no-mask' tasks where tasks contain no masking. 2) 'high-mask' tasks where tasks have significant potion (50%) of masking [2]. All experiment settings are run in 3 seeds. Results are shown in Figure 9.

4.5.1 Dataset Corruption

We firstly test how an ill-collected dataset would affect performance of RoBERT. To do so, we deliberately corrupt collected dataset by replacing certain proportion of its trajectories by a randomly picked trajectory. Then we perform normal training over these corrupted dataset and obtain results in Figure 9a.

From Figure 9a, we find: 1) On both no masking and high masking tasks, low corruption ratio doesn't significantly hinder performance of RoBERT. 2) For high corruption ratio ($\geq$ 30%), the decrease of RoBERT's performance is still in the same magnitude compared to the original one. Recorded video of RoBERT trained on corruption ratio 45% also shows no significant performance degradation, reflecting RoBERT's robustness towards a skewed dataset.

[2] We also add task 'rock-scissor-paper-high-mask' and 'fire-handgun-high-mask' where mask ratio is deliberately increased for better evaluation.



High masking

(a) corrupted dataset

(b) masking strategy

(c) #. of action bins

No masking

Figure 9: Analysis for different settings of RoBERT. Errorbar represents 95% confidence interval.

4.5.2 Masking Strategy

In this section, we conduct experiment to test how different masking strategy affect model training. We select 6 masking strategy: 1. CL-based masking used in main results. ('CL'). 2. fixed masking ratio (15%, 35%, 50%, 70%) 3. no masking at all ('No').

Results are shown in Figure 9b. What we observe is: 1. No masking, as expected, leads to worst performance on high masking tasks but best performance on no-masking tasks. 2. static masking of low ratio could already achieve decent performance compared to CL-based one.

We also record videos for no masking and 15% static masking ratio. Results suggest: 1) no masking, 15% masking and default CL masking obtain similar precision over unmasked keyframes. 2) for masked frames, 'no masking' results in worst precision while 15% static masking performs best and default setting in between.

4.5.3 Number (#) of Action Bins

As in Section 3.3, we also tried the method in Shafiullah et al. (2022) to tackle multi-modality situation introduced by keyframe-based control. Herein, we experiment over various choices of number of action bins (1, 2, 5, 10 (default), 32, 48, 64 and 96) and obtain the result in Figure 9c.

We find: 1) Large number of action bins would increase performance for high-masking tasks but hinder performance on no-masking tasks. 2) Surprisingly, no multi-modality (action bins = 1) already achieves decent performance, compared to multi-modal version.

We also record videos for action bins 1 and 2, and find: 1) Only 1 action bin leads to most stable behavior generation without serious performance degradation 2) Using 2 action bins slightly introduces 'shaking' in action generation but still smaller than default. This shows that setting action bins to small values would generally enable control both stably and precisely.

5 Summary

In this paper, we present RoBERT, a low-cost method to obtain bi-directional sequence models for *zero-shot*, *multi-task* and *keyframe-based* robot control. Extensive experiments and analysis are conducted, to demonstrate that owing to recent progress in unsupervise RL and computing ability, RoBERT could be obtained with minimum human involvement and serve as a robust foundation model for further improvements. We believe in the field of LLM-based Robotics, the accessibility of RoBERT would enable more researches and participations that benefits the whole community.

REFERENCES

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control, July 2023a. URL `http://arxiv.org/abs/2307.15818`. arXiv:2307.15818 [cs].

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale, August 2023b. URL `http://arxiv.org/abs/2212.06817`. arXiv:2212.06817 [cs].

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in neural information processing systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-Scale Study of Curiosity-Driven Learning. September 2018a. URL `https://openreview.net/forum?id=rJNwDjAqYX`.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018b.

Jinglin Chen, Aditya Modi, Akshay Krishnamurthy, Nan Jiang, and Alekh Agarwal. On the Statistical Efficiency of Reward-Free Exploration in Non-Linear RL. May 2022. URL `https://openreview.net/forum?id=65eqtvEShR8`.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision Transformer: Reinforcement Learning via Sequence Modeling. November 2021. URL `https://openreview.net/forum?id=a7APmM4B9d`.

Xin Chen, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, and Gang Yu. Executing your commands via motion diffusion in latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18000–18010, 2023.

Suyoung Choi, Gwanghyeon Ji, Jeongsoo Park, Hyeongjun Kim, Juhyeok Mun, Jeong Hyun Lee, and Jemin Hwangbo. Learning quadrupedal locomotion on deformable terrain. *Science Robotics*, 8(74):eade2256, 2023. doi: 10.1126/scirobotics.ade2256. URL `https://www.science.org/doi/abs/10.1126/scirobotics.ade2256`. tex.eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.ade2256.

MuJoCo Menagerie Contributors. MuJoCo Menagerie: A collection of high-quality simulation models for MuJoCo, 2022. URL http://github.com/google-deepmind/mujoco_menagerie.

Leilei Cui, Shuai Wang, Jingfan Zhang, Dongsheng Zhang, Jie Lai, Yu Zheng, Zhengyou Zhang, and Zhong-Ping Jiang. Learning-based balance control of wheel-legged robots. *IEEE Robotics and Automation Letters*, 6(4):7667–7674, 2021. doi: 10.1109/LRA.2021.3100269.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is All You Need: Learning Skills without a Reward Function. September 2018. URL https://openreview.net/forum?id=SJx63jRqFm.

Hiroki Furuta, Yutaka Matsuo, and Shixiang Shane Gu. Generalized Decision Transformer for Offline Hindsight Information Matching. October 2021. URL https://openreview.net/forum?id=CAjxVodl_v.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DEBERTA: DECODING-ENHANCED BERT WITH DISENTANGLED ATTENTION. October 2020. URL https://openreview.net/forum?id=XPZIaotutsD.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models. August 2023. URL https://openreview.net/forum?id=9_8LF30mOC&referrer=%5Bthe%20profile%20of%20Jiajun%20Wu%5D(%2Fprofile%3Fid%3D~Jiajun_Wu1).

Michael Janner, Qiyang Li, and Sergey Levine. Offline Reinforcement Learning as One Big Sequence Modeling Problem. November 2021. URL https://openreview.net/forum?id=wgeK563QgSw.

Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-Free Exploration for Reinforcement Learning. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 4870–4879. PMLR, November 2020. URL https://proceedings.mlr.press/v119/jin20d.html. ISSN: 2640-3498.

Changyeon Kim, Jongjin Park, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. Preference transformer: Modeling human preferences using transformers for RL. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Peot1SFDX0.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

Kuang-Huei Lee, Ofir Nachum, Sherry Yang, Lisa Lee, C. Daniel Freeman, Sergio Guadarrama, Ian Fischer, Winnie Xu, Eric Jang, Henryk Michalewski, and Igor Mordatch. Multi-game decision transformers. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=0gouO5saq6K.

Fangchen Liu, Hao Liu, Aditya Grover, and Pieter Abbeel. Masked Autoencoding for Scalable and Generalizable Decision Making. October 2022. URL https://openreview.net/forum?id=lNokkSaUbfV.

Hao Liu and Pieter Abbeel. APS: Active Pretraining with Successor Features. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 6736–6747. PMLR, July 2021a. URL https://proceedings.mlr.press/v139/liu21b.html. ISSN: 2640-3498.

Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. *Advances in Neural Information Processing Systems*, 34:18459–18473, 2021b.

Sam Lobel, Akhil Bagaria, and George Konidaris. Flipping coins to estimate pseudocounts for exploration in reinforcement learning. In *International Conference on Machine Learning*, 2023.

Shangke Lyu and Chien Chern Cheah. Data-driven learning for robot control with unknown Jacobian. *Automatica*, 120:109120, 2020. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2020.109120. URL https://www.sciencedirect.com/science/article/pii/S0005109820303186.

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, 2022. doi: 10.1126/scirobotics.abk2822. URL https://www.science.org/doi/abs/10.1126/scirobotics.abk2822. tex.eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.abk2822.

I. Made Aswin Nahrendra, Byeongho Yu, and Hyun Myung. DreamWaQ: Learning Robust Quadrupedal Locomotion With Implicit Terrain Imagination via Deep Reinforcement Learning, March 2023. URL http://arxiv.org/abs/2301.10602. arXiv:2301.10602 [cs, eess].

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. *CoRR*, abs/1705.05363, 2017. URL http://arxiv.org/abs/1705.05363.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. DeepMimic: example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics*, 37(4):143:1–143:14, July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201311. URL https://dl.acm.org/doi/10.1145/3197517.3201311.

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. ASE: large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions on Graphics*, 41(4):94:1–94:17, July 2022. ISSN 0730-0301. doi: 10.1145/3528223.3530110. URL https://dl.acm.org/doi/10.1145/3528223.3530110.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-maron, Mai Giménez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=1ikK0kHjvj. Featured Certification, Outstanding Certification.

Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior Transformers: Cloning k modes with one stone. October 2022. URL https://openreview.net/forum?id=agTr-vRQsa.

Jean Tarbouriech, Matteo Pirotta, Michal Valko, and A. Lazaric. Reward-Free Exploration beyond Finite-Horizon. 2020. URL https://www.semanticscholar.org/paper/Reward-Free-Exploration-beyond-Finite-Horizon-Tarbouriech-Pirotta/43e44493aa061a9c96ae66907e3a33d68e025943.

Chen Tessler, Yoni Kasten, Yunrong Guo, Shie Mannor, Gal Chechik, and Xue Bin Peng. CALM: Conditional Adversarial Latent Models for Directable Virtual Characters. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, pp. 1–9, New York, NY, USA, July 2023. Association for Computing Machinery. ISBN 9798400701597. doi: 10.1145/3588432.3591541. URL https://dl.acm.org/doi/10.1145/3588432.3591541.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Xin Wang, Yudong Chen, and Wenwu Zhu. A Survey on Curriculum Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576, September 2022. ISSN 1939-3539. doi: 10.1109/TPAMI.2021.3069908. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Philipp Wu, Arjun Majumdar, Kevin Stone, Yixin Lin, Igor Mordatch, Pieter Abbeel, and Aravind Rajeswaran. Masked Trajectory Models for Prediction, Representation, and Control. March 2023. URL https://openreview.net/forum?id=tT3LUdmzbd.

Chuanyu Yang, Kai Yuan, Qiuguo Zhu, Wanming Yu, and Zhibin Li. Multi-expert learning of adaptive legged locomotion. *Science Robotics*, 5(49):eabb2174, 2020. doi: 10.1126/scirobotics.abb2174. URL https://www.science.org/doi/abs/10.1126/scirobotics.abb2174. tex.eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.abb2174.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Reinforcement learning with prototypical representations. In *International Conference on Machine Learning*, pp. 11920–11931. PMLR, 2021.

Denis Yarats, David Brandfonbrener, Hao Liu, Michael Laskin, Pieter Abbeel, Alessandro Lazaric, and Lerrel Pinto. Don't Change the Algorithm, Change the Data: Exploratory Data for Offline Reinforcement Learning, April 2022. URL http://arxiv.org/abs/2201.13425. arXiv:2201.13425 [cs].

A Appendix

A.1 Details of Main Experiments

All experiments are conducted in Mujoco simulator (Todorov et al., 2012). Model specifications of UR5E and Shadow are adopted from mujoco_menagerie (Contributors, 2022). We use a control rate of 20Hz for all envs. In all envs., model output is directly applied as action signal (force or torque) without any further processing.

For all experiments, we use a length of history window of 30 frames (1.5s) and length of command window of 20 frames (1s).

To collect dataset, we train unsupervise RL algorithm (namely, RND (Burda et al., 2018b), APT (Liu & Abbeel, 2021b), APS (Liu & Abbeel, 2021a), ICM (Pathak et al., 2017) and Proto (Yarats et al., 2021)) for some large total steps and record network snapshot after a small step. Then we use these snapshots to rollout in environment, collecting trajectories of state-actions as dataset. At last, the datasets are merged together to form the final dataset used in each environment. Usually we tend to set the total step as a large number *rule-of-thumb*-ly and it works quite well. But for UR5E where unsupervise RL algorithm tends to collect extreme behaviors, we stop data collecting phases relatively early and this is the only tuning we performed.

For Pointmass and ShadowHand, we extract 18 (6 unmasked + 6 low-masked + 6 high-masked) trajectories from test set. For UR5E, we extract 90 (30 unmasked + 30 low-masked + 30 high-masked) trajectories. Then these trajectories are merged with manual-designed tasks (15, 4, 18 for Pointmass, UR5E and ShadowHand, respectively).

For test tasks used during training phase, we restrict their length to be 110 frames (30 for history window + 80 for commands length) as a constraint of time and computing resources.

We tune learning rate with a linear warm up and cosine annealing as in Brown et al. (2020).

The CL-based masking starts from 0 and gradually increases to 'max' (only 1 frame is unmasked).

A.1.1 Details of Scaling Effects

To experiment over scaling effects on pointmass, we first collect a large dataset (304M transitions) then shrink it to smaller size by sub-selecting trajectories with a fixed step and adjust learning rate and batch size slightly to fit for small-sized dataset. We run 3 seeds for each setting and random policy then report mean of their performance in Figure 3.

A.2 Default Hyperparameters

We list default settings used for model training in Pointmass, UR5E and ShadowHand environments in Table 2.

Table 2: Default hyperparameters used in model training

Hyperparameter	Value	Remarks
Learning Rate	7e-5	learning rate without scheduling
Total Training Frames	5.12M (Pointmass) 6.144M (UR5E) 7.68M (ShadowHan)	
Learning Rate Warm-up	10%	linear warm-up from 0.01% to 100%
Cosine Annealing begins	35%	cosine annealing from 100% to 0.1%
Cosines Annealing ends	85%	
Batch Size	512	
Hidden Dim. of Transformer	768	
Num. Layers of Transformer	6	
Num. Head of Transformer	6	
Num. of Action bins	10	
Masking begins	2%	linear increasing mask ratio
Masking ends	62%	

A.3 Details of Analysis Experiments

We use the same setting of main experiments for analysis experiments.

A.4 List of Contents in Supplement Materials

We have uploaded supplement materials and list contents of it here as a quick reference:

- code: This is training/evaluating code for RoBERT as used to produce results in this paper.
- videos: all video results:
  - main_result: these are main results for Pointmass, UR5E and Shadow Hand environments.
  - pointmass_shrink_5184: these are task fulfillment results for RoBERT trained in shrinked pointmass dataset.
  - analysis: results for Analysis section:
    - shadow_default: This is model trained with default setting on tasks of analysis section. This is mainly for reference purpose.
    - shadow_corrupt_0.45: This is model trained with corrupted dataset.
    - shadow_mask_15: This is results of model trained with static 15% masking.
    - shadow_mask_no: This is results of model trained with no masking applied.
    - shaodw_num_center_1: This is results of model trained with only 1 action bin.
    - shdow_num_center_2: This is results of model trained with 2 action bins.