Emergent Agentic Transformer from Chain of Hindsight Experience

Hao Liu Pieter Abbeel
University of California, Berkeley

Abstract

Large transformer models powered by diverse data and model scale have dominated natural language modeling and computer vision and pushed the frontier of multiple AI areas. In reinforcement learning (RL), despite many efforts into transformer-based policies, a key limitation, however, is that current transformer-based policies cannot learn by directly combining information from multiple sub-optimal trials. In this work, we address this issue using recently proposed chain of hindsight to relabel experience, where we train a transformer on a sequence of trajectory experience ascending sorted according to their total rewards. Our method consists of relabelling target return of each trajectory to the maximum total reward among in sequence of trajectories and training an autoregressive model to predict actions conditioning on past states, actions, rewards, target returns, and task completion tokens, the resulting model, Agentic Transformer (AT), can learn to improve upon itself both at training and test time. As we show on D4RL and ExoRL benchmarks, to the best our knowledge, this is the first time that a simple transformer-based model performs competitively with both temporal-difference and imitation-learning-based approaches, even from sub-optimal data. Our Agentic Transformer also shows a promising scaling trend that bigger models consistently improve results.

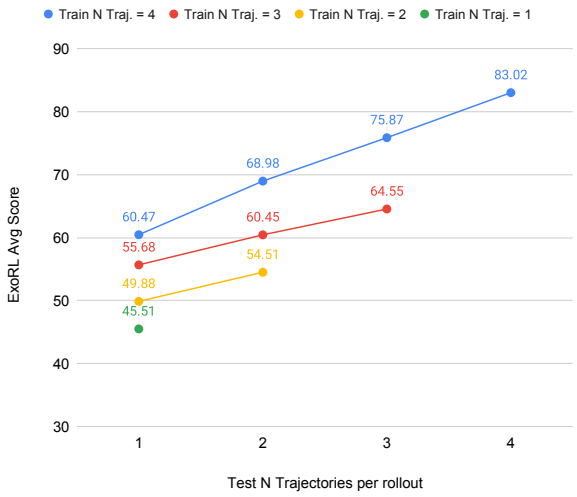

Figure 1. Agentic Transformer can automatically improve its performance at evaluation time by rollouting more trajectories in a trial-and-error manner. The scaling improves with both more chain of hindsight training sequences.

1. Introduction

Large transformer (Vaswani et al., 2017) models have substantially advanced the state-of-the-art across a variety of domains, including natural language processing tasks (Devlin et al., 2018; Brown et al., 2020; Liu et al., 2019), computer vision (Dosovitskiy et al., 2020; Alayrac et al., 2022), and code generation (Lewkowycz et al., 2022; Chen et al., 2021c).

Despite the successes, a key limitation is that these models are not agentic, *i.e.* they cannot interact with the real world to accomplish tasks like a robot. Reinforcement learning (RL), on the other hand, in principle is designed for building interactive agents. However, conventional RL algorithms are limited to small models (*e.g.*, an MLP with two layers) and are difficult to train and scale (see e.g. Andrychowicz et al., 2020). The difficulty of scaling the model size in conventional RL algorithms make it difficult to take advantage of large Transformer models.

In order to combine Transformer with decision-making, there have been lots of efforts in attempting to cast RL from offline data as a sequence modeling problem (Chen et al., 2021a; Laskin et al., 2022; Reed et al., 2022). For instance, DT (Chen et al., 2021a) proposes to train a Transformer to autoregressively predict action sequences based on sequences of returns-to-go and states.

Despite the progress made, existing Transformer based

Correspondence to: Hao Liu <hao.liu@cs.berkeley.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

decision-making models cannot learn by directly combining information from multiple sub-optimal trials, in fact, they require high-return data to achieve high return (see e.g. Chen et al., 2021a; Laskin et al., 2022; Yarats et al., 2022), indicating the lack of extrapolation ability besides the imitation learning ability. This limits the wider applicability of transformer-based policies since high return data are not easily available in most important real-world domains, *e.g.*, health care and industry robots.

To resolve these issues, we first hypothesize that the fact that existing Transformer based decision-making models underperform TD-learning approaches and lack of extrapolation is due to the fact that during training and inference, the model can only do one trial. Our key observation is that one ability humans have, unlike the current generation of models, is to learn almost as much from achieving an undesired outcome as from the desired one. We take the approach *chain of hindsight* introduced in Liu et al. (2023) which proposes to condition language model on positive indicator and negative example to predict positive example, and vice versa. The idea applies to learn decision making – imagine learning basketball and attempting a shot that misses the net on the right. Existing models conclude that the sequence of performed actions don't result in success, and little is learned. It is however possible to chain another attempt's sequence of actions which missed even more far away with this sequence of actions, as if this sequence of actions would be a successful second attempt if the goal is placing the ball closer to the net.

In this paper, we propose to train Transformer to perform exactly this kind of reasoning. Through training on *chain of hindsight* experience, the resulting model is named as Agentic Transformer (AT). Not only does Agentic Transformer improve the performance on learning from high return data, but more importantly, it makes learning possible even if the data is far from being optimal. Our approach is based on training a decoder-only Transformer (Radford et al., 2018; 2019; Brown et al., 2020) which takes as input not only the current episode, but also multiple episodes whose returns are lower than current episode's return and are ascending sorted according to their returns. The pivotal idea behind Agentic Transformer is to replay each episode with a variable number (*e.g.*, randomly choose between 0 and 4) of episodes to form chain of hindsight experience, as if the model was trying to improve from previous episode(s) to current episode.

Agentic Transformer achieves state-of-the-arts on standard RL benchmarks including D4RL (Fu et al., 2020) and ExoRL (Laskin et al., 2021; Yarats et al., 2022). Agentic Transformer can learn by directly combining information from multiple sub-optimal trials and being able to improve itself through multiple trials at test time. Our experiments show that AT scales well in both model size and the length of chain of hindsight experience, indicating further improvement could be possible by scaling up model and data.

2. Preliminaries

2.1. Reinforcement Learning

We consider learning problem in the context of a Markov Decision Process (MDP) represented by the tuple ($\mathcal{S}$, $\mathcal{A}$, P, $\mathcal{R}$). The MDP tuple consists of states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, transition dynamics $P(s'|s,a)$, and a reward function $r = \mathcal{R}(s,a)$. To describe the state, action, and reward at time step t, the notations s_t, a_t, and $r_t = \mathcal{R}(s_t, a_t)$ are used. A trajectory is a sequence of states, actions, and rewards and is denoted by $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T)$. The return of a trajectory at time step t, $R_t = \sum_{t'=t}^{T} r_{t'}$, is calculated as the sum of future rewards from that time step. The goal of reinforcement learning is to find a policy that maximizes the expected return $\mathbb{E}\left[\sum_{t=1}^{T} r_t\right]$ in an MDP. In supervised or offline reinforcement learning, data is obtained from a fixed limited dataset of trajectory rollouts from arbitrary policies, instead of from environment interactions. This setting eliminates the ability of the agents to explore the environment and gather additional feedback. Conventional datasets either consist mainly of high quality, near optimal trajectories like in D4RL (Fu et al., 2020) which are obtained by running trained expert policies or by storing the experience of training an expert policy, or mainly consist of diverse, exploratory and sub-optimal trajectories like in ExoRL (Yarats et al., 2022) where trajectories are collected through unsupervised exploration algorithms.

2.2. Transformers

The Transformer (Vaswani et al., 2017) architecture consists of multiple layers of self-attention operation and MLP. The self-attention begins by projecting input data X with three separate matrices onto D-dimensional vectors called queries Q, keys K, and values V. These vectors are then passed through the attention function:

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T/\sqrt{D})V. \quad (1)$$

The QK^T term computes an inner product between two projections of the input data X. The inner product is then normalized and projected back to a D-dimensional vector with the scaling term V. Transformers (Vaswani et al., 2017; Devlin et al., 2018; Brown et al., 2020) utilize self-attention as a core part of the architecture to process sequential data such as text sequences. Transformers are usually pre-trained with a self-supervised objective. Common prediction tasks include predicting randomly masked out tokens (Devlin et al., 2018) or applying a causal mask and predicting the next token (Radford et al., 2018; Brown et al., 2020). The

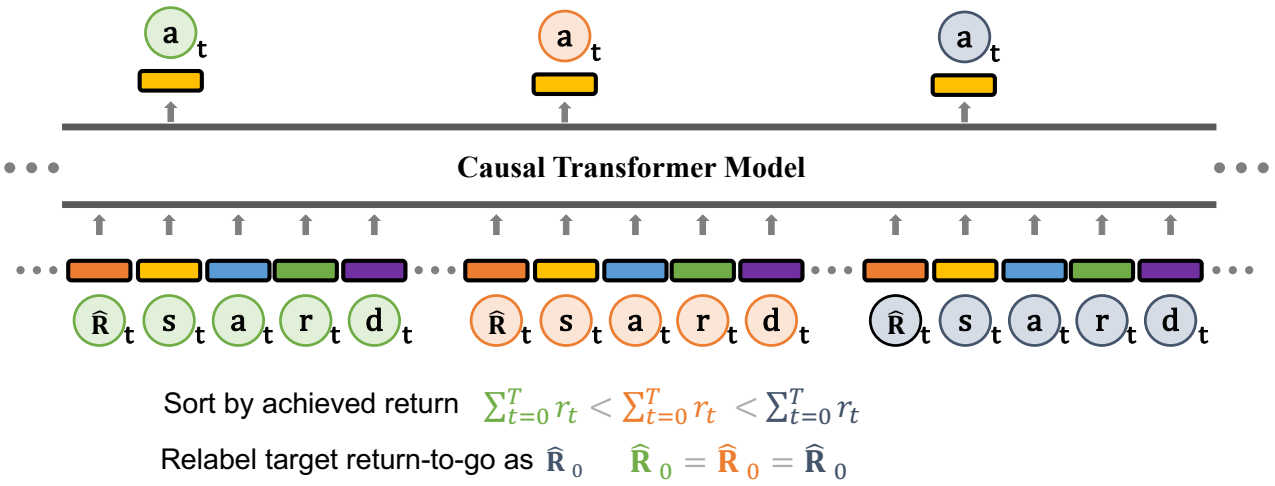

Figure 2. **Agentic Transformer**. The input sequence consists of multiple episodes ascending sorted according to their total rewards. The initial desired return $\widehat{R}_0$ of all trajectories are set to the maximum total rewards among all trajectories. For each trajectory, the return-to-go is updated using rewards in the same trajectory: $\widehat{R}_t = \widehat{R}_0 - \sum_{j=0}^{t} r_j$. The task completion token d indicates whether achieved cumulative rewards in a trajectory is larger than desired target return(Equation 2), this gives model feedback on past trajectories and help steer model to try to reach target return in next trajectory at test time. States, actions, rewards, returns-to-go, and task completion are fed into modality specific linear embeddings and a positional episodic timestep encoding is added. Tokens are fed into a GPT architecture which predicts actions autoregressively using a causal self-attention mask. At `training time`: The model is trained to predict action tokens in the last (best) trajectory conditioning on past trajectories, states, actions, returns-to-go and task completion tokens. At `testing time`: The model predicts action autoregressively across multiple trajectories.

GPT architecture (Radford et al., 2018) replaces the summation/softmax over the n tokens with only the previous tokens in the sequence ($j \in [1, i]$), enabling autoregressive generation by using causal self-attention mask. In this work, we use the GPT architecture because we need to do autoregressive generation at test time.

2.3. Transformer based Behavior Cloning

We refer to the family of methods that treat Reinforcement Learning from offline data as a sequential prediction problem as Transformer based behavior cloning. Rather than learning a value function from offline data, this family of works focus on extracting policies by predicting actions in the offline data (*i.e.* behavior cloning) with an autoregressive sequence model and either return conditioning (Chen et al., 2021b; Laskin et al., 2022; Lee et al., 2022) or filtering out suboptimal data (Reed et al., 2022) or training masked sequence model bypredicting masked states and actions tokens (Liu et al., 2022a; Carroll et al., 2022).

3. Method

In this section, we present Agentic Transformer (AT), which models chain of hindsight experience trajectories autoregressively based on Transformer archiecture, as summarized in Figure 2 and Algorithm 1.

Chain of hindsight Experience. The key factors that influenced our decision on how to represent trajectories are: (1) the ability of transformers to uncover meaningful patterns from multiple trajectories sampled from arbitrary offline data, and (2) the capacity to produce actions conditionally during evaluation and improve itself conditions on collected experience. Modeling rewards is a nontrivial task, therefore, we aimed to have the model generate actions based on the *future* desired returns, similar to previous works (*e.g.*, Chen et al., 2021a; Laskin et al., 2022), rather than relying on past rewards. We feed the model with the initial target returns-to-go $\widehat{R}_0$ and update $\widehat{R}_t = \widehat{R}_0 - \sum_{j=0}^{t} r_j$ using rewards. We also feed the model with a completion token d that indicates whether the achieved cumulative rewards in a trajectory are larger than or equal to desired returns-to-go, specifically

$$d_T = \mathbb{1}\left(\sum_{j=0}^{T} r_j \geq \widehat{R}_0\right) \quad d_i = 0,\ \forall i \in [1, T-1], \quad (2)$$

where $\mathbb{1}$ is indicator function. This leads to the following trajectory representation which is amenable to autoregressive training and generation:

$$\tau = \left(\widehat{R}_0, s_0, a_0, r_0, d_0, \ldots, \widehat{R}_T, s_T, a_T, r_T, d_T\right)$$
$$\text{where} \quad \widehat{R}_t = \widehat{R}_0 - \sum_{j=0}^{t} r_j. \quad (3)$$

Since we want the model to learn to 'stitch' sub-optimal data rather than just imitating optimal data, and at test time we want the model to achieve desired target return through multiple trajectories of trial-and-errors, we construct a chain of hindsight experience for the model to learn to improve even from sub-optimal data and learning to self-improve during test time. To achieve this, we take the approach called *chain of hindsight* (Liu et al., 2023) which trains language model from human feedback by conditioning on positive indicator and negative rated example to predict corresponding positive rated example. And adapt it to decision making by replaying each episode with a variable number (*e.g.*, randomly choose between 0 and 4) of episodes to form *chain of hindsight* experience, as if the model was trying to improve from previous episode(s) to current episode.

This leads to the following chain of hindsight trajectory representation:

$$s = \left(\tau^1, \tau^2 \ldots, \tau^n\right) \tag{4}$$

where

$$\tau^i = \left(\widehat{R}^i_0, s^i_0, a^i_0, r^i_0, d^i_0, \ldots, \widehat{R}^i_T, s^i_T, a^i_T, r^i_T, d^i_T\right) \tag{5}$$

$s.t.$

$$\sum_{t=1}^{T} r^0_t \leq \sum_{t=1}^{T} r^1_t \leq \cdots \leq \sum_{t=1}^{T} r^n_t \tag{6}$$

$$\widehat{R}^i_0 = \sum_{t=1}^{T} r^n_t \quad \forall\, 1 \leq i \leq n \tag{7}$$

$$\widehat{R}^i_t = \widehat{R}^i_0 - \sum_{j=0}^{t} r^i_j \quad \forall\, 1 \leq i \leq n, \tag{8}$$

Equation 6 states the ordering requirement, meaning that trajectories are ascending sorted according to their total reward. Equation 7 sets the *hindsight* target: for all n trajectories, initial target equals to trajectory n's total reward. Equation 8 updates returns-to-go using trajectory reward.

At test time, we can specify the desired performance (e.g. 1 for success or 0 for failure), as well as the environment starting state, and the conditioning information to initiate generation. After executing the generated action for the current state, we decrement the target return by the achieved reward and repeat until episode termination. If the target return is not achieved, the model starts a new episode and continues interacting with the environment until the maximum episode number is reached.

Architecture. We feed the n trajectories into Agentic Transformer, this results in a total of $5 \times n \times T$ tokens, with one token for each of the five modalities: returns-to-go, state, action, reward, and completion. To create the token embeddings, a linear layer is trained for each modality which transforms the raw inputs into the desired embedding dimension, followed by layer normalization (Ba et al., 2016). In addition to this, an embedding for each time step is also learned and added to the tokens, which is distinct from the standard positional embedding used in transformers where one time step is represented by five tokens. Finally, the tokens are processed by a GPT model (Radford et al., 2018) that predicts future action tokens through autoregressive modeling.

Training and Test. We are given a dataset of offline trajectories. We sample minibatches of trajectories from the dataset. The model predicts the action token a_t given the input token s_t, and the prediction is evaluated with either cross-entropy loss or mean-squared error, depending on whether the actions are discrete or continuous. The losses from each time step are averaged. Note that only the action tokens a_t from the last trajectory τ^n are used for loss calculation. While it's feasible to predict other tokens or use other trajectories in the training process, we didn't observe improvements in performance and consider it as a potential area for future research. At test time, following standard practice in NLP, we cache key and query during autoregressive decoding to speed up inference. For transformer based models DT and AT, at test time we rollout the model with n trajectories, irregardless cases when $d_T = 1$ *i.e.* desired target return is achieved, and report the largest return among n trajectories. For DT the maximum return is achieved at the 1st trajectory while AT improves itself along the trajectory sequence and achieves higher return with more trajectories. The model sizes are shown in Table 1, base is used by default unless otherwise mentioned. Since in our default configuration $n = 4$, and T is typically 1000 in D4RL and ExoRL, total sequence length is 20,000 which uses a large amount of memory for large models. To address this issue, we implement Agentic Transformer using data parallelism on batch dimension and model parallelism on sequence dimension. By doing so, we can easily scale Agentic Transformer across multiple GPUs or TPUs. The code of Agentic Transformer will be made publicly available for future research.

Algorithm 1 Training Agentic Transformer

Required: Dataset of Trajectories, Transformer Model
Required: Max Iterations m, Max Number of trajectories in chain of hindsight experience n
Initialize
for $i = 1$ **to** $m - 1$ **do**
  Randomly sample j from 1 to n
  Randomly sample j episodes from dataset
  Compute returns-to-go $\hat{R}$ for all steps for each episode
  Sort j episodes ascending according to their returns
  Let $\hat{R}_{\text{max}}$ be the return of the last episode
  For each other episode, recomputing its returns-to-go by setting $\hat{R}_0 = \hat{R}_{\text{max}}$
  Concatenate j episodes as a sequence
  Train Transformer to predict next action token (see Figure 2).
end for

Model	Layers	# of heads	d_{model}	Batch size
Small	2	4	64	256
Base	4	8	256	256
Large	6	16	512	256
XLarge	8	16	512	256

Table 1. Architecture details of different sized models used in Agentic Transformer. We list the number of layers, d_{model}, the number of attention heads and attention head size, training batch size, and sequence length. The feed-forward size d_{ff} is always $4 \times d_{\text{model}}$ and attention head size is always 16.

4. Experiments

Dataset: D4RL. In this section, we consider the continuous control tasks from the D4RL benchmark (Fu et al., 2020). The different dataset settings are described below.

1. Medium: 1 million timesteps generated by a "medium" policy that performs approximately one-third as well as an expert policy.
2. Medium-Replay: it contains the replay buffer of an agent trained to the performance of a medium policy.
3. Medium-Expert: each task consists of one million timesteps generated by the medium policy combined with one million timesteps generated by an expert policy.

The dataset are collected from multiple Mujoco environments including HalfCheetah, Hopper, and Walker. Since D4RL dataset is collected by conventional RL algorithms, it consists of many high return trajectories that are near expert. Therefore, filtered behavior cloning (*e.g.*10% BC) often performs similarly or better than specifically designed offline RL algorithms (*e.g.*DT). In order to evaluate our method in a more challenging and realistic setting, we consider ExoRL (Yarats et al., 2022) dataset that only consists of diverse and low return trajectories.

Dataset: ExoRL. The ExoRL dataset is based on unlabeled exploratory data collected by running unsupervised RL algorithms. For each environment, it comes with eight different unsupervised data collection algorithms, taken from from URLB (Laskin et al., 2021). The datasets are collected by unsupervised RL and then relabeled using task reward function. In light of the benefit of scaling up data (Hoffmann et al., 2022), we opted to use the combination of all datasets for all baselines and our method. Specifically, for each environment, we combine the datasets collected by eight algorithms (Pathak et al., 2017; 2019; Burda et al., 2019; Liu & Abbeel, 2021b; Yarats et al., 2021; Eysenbach et al., 2019; Lee et al., 2019; Liu & Abbeel, 2021a). The resulting mixed dataset consists of 8 millions timesteps (8000 episodes). Since it is collected by unsupervised RL without using task rewards, the dataset is optimized for diversity but is far from optimal task rewards. The details are referred to the original papers.

Baselines. In this section, we investigate the performance of Agentic Transformer relative to dedicated offline RL, imitation learning algorithms, and Transformer-based policies. In particular, our primary points of comparison are prior Transformer-based policies such as decision transformer since architecture wise Agentic Transformer is similar them. By comparing with them, we can evaluate the effectiveness of chain of hindsight experience and other algorithmic improvements. We further compare with model-free offline RL algorithms based on TD-learning, since architecture is fundamentally model-free in nature as well. Furthermore, TD-learning is the dominant paradigm in RL for sample efficiency and is effective at learning from sub-optimal data. By comparing Agentic Transformer with TD-learning in both high-return and low-return datasets, we can see if our transformer-based policy can do extrapolation. We also compare with behavior cloning and variants, since it also involves a likelihood based policy learning formulation similar to ours. Our baselines can be categorized as follows:

- **Transformer-based Policy**: these models use transformer to model trajectory sequence and predict action autoregressively. We consider decision transformer (DT) (Chen et al., 2021b) which is shown to be effective on D4RL.
- **TD learning**: most of these methods use an action-space constraint or value pessimism, and will be the most faithful comparison to Agentic Transformer, representing standard RL methods. We consider state-of-the-art TD3+BC (Fujimoto & Gu, 2021) which is shown to be effective on D4RL and TD3 (Fujimoto et al., 2018) which is shown to be effective on ExoRL.
- **Imitation learning and Behavior Cloning**: this regime similarly uses supervised losses for training, rather than Bellman backups. We consider BC-10%. BC-10% is shown to be competitive to state-of-the-arts on D4RL. DT also belongs to this category since it is a transformer based return conditioned BC, both are closely related to our model.

In total for offline RL we use five algorithms: BC-10%, TD3+BC, TD3 and DT. We adhere closely to the original hyper-parameter settings for each algorithm, but in several cases we perform hyper-parameter tuning to achieve best possible performance. We train offline RL algorithms for 500k gradient updates and then evaluate by rolling out 10 episodes in the environment. We report mean and standard error across 3 random seeds.

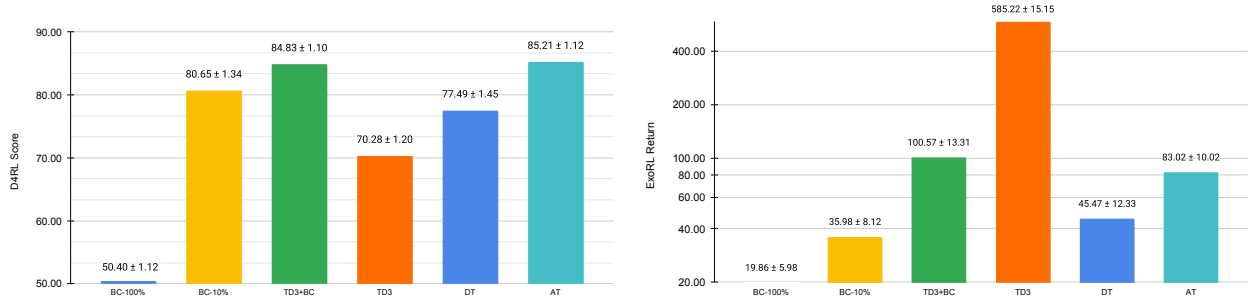

Figure 3. Agentic Transformer performs competitively with both temporal-difference based and imitation-learning based approaches in ExoRL as well as D4RL tasks. **Left**. Tasks average performance on D4RL. **Right**. Tasks average performance on ExoRL. We report the mean and variance for three seeds.

Table 2. Results for D4RL datasets. We report the mean and variance for three seeds. Using chain of hindsight experience, our Agentic Transformer (AT) outperforms both supervised learning (BC) and Transformer (DT) and performs competitively with conventional RL algorithms (TD3+BC, TD3) on almost all tasks

Dataset	**Environment**	BC-10%	TD3+BC	TD3	DT	Agentic Transformer (AT)
Medium-Expert	HalfCheetah	94.11	**96.59**	87.60	93.40	95.81 ± 0.25
Medium-Expert	Hopper	113.13	113.22	98.41	111.18	**115.92 ± 1.26**
Medium-Expert	Walker	109.90	112.21	100.52	108.71	**114.87 ± 0.56**
Medium	HalfCheetah	43.90	**48.93**	34.60	42.73	45.12 ± 0.34
Medium	Hopper	73.84	**70.44**	56.98	69.42	**70.45 ± 0.45**
Medium	Walker	82.05	86.91	70.95	74.70	**88.71 ± 0.55**
Medium-Replay	HalfCheetah	42.27	45.84	38.81	40.31	**46.86 ± 0.33**
Medium-Replay	Hopper	90.57	**98.12**	78.90	88.74	96.85 ± 0.41
Medium-Replay	Walker	76.09	91.17	65.94	68.22	**92.32 ± 1.21**
Total Average		80.65	84.83	70.30	77.49	**85.21**

4.1. D4RL results

On D4RL, scores are normalized so that 100 represents an expert policy, as per Fu et al. (2020). Baselines numbers are reported by the original papers and from the D4RL paper. Agentic Transformer surpasses the baselines in a wide range of tasks. Our results are shown in Table 2. Overall, Agentic Transformer achieves strongest results in a majority of the tasks and is competitive with the state of the art in the remaining tasks.

Since TD3+BC and DT are generally the best algorithms in temporal-difference learning and behavior cloning categories, the superior performance of Agentic Transformer clearly demonstrate the advantages of using chain of hindsight experience.

4.2. ExoRL results

On ExoRL, we report the cumulative return, as per Yarats et al. (2022). BC, TD3+BC, and TD3 numbers are from the ExoRL paper, DT numbers are run by ourselves. Our results are shown in Table 3. Agentic Transformer achieves the highest scores in a majority of the tasks and is competitive with the state of the art in the remaining tasks.

Since the ExoRL data is significantly more diverse than D4RL because it is collected using unsupervised RL (Laskin et al., 2021), it is found that temporal-difference learning performs best while behavior cloning struggles. Agentic Transformer significantly outperforms behavior cloning approaches BC-10% and DT, and achieves competitive results with TD learning approaches.

We further evaluate Agentic Transformer with different models sizes. We select two tasks from ExoRL in order to reduce compute cost incurred by XLarge model size. Figure 4 shows the results. Agentic Transformer improves with larger model size, showing promising scaling behavior.

4.3. Evaluation of Agency

At test time, the total rewards of each trajectory in a sequence are reported in Figure 1. We follow DT's experimental settings and use their target return as initial return-to-go for both DT and AT.

Table 3. Results for ExoRL datasets. We report the mean and variance for three seeds. Using chain of hindsight experience, our Agentic Transformer (AT) outperforms both supervised learning (BC) and Transformer (DT) on almost all tasks, and performs competitively with conventional RL algorithms (TD3+BC, TD3).

Dataset	**Task**	BC-10%	TD3+BC	TD3	DT	Agentic Transformer (AT)
All	Walker Stand	52.91	67.13	832.10	34.54	68.55
All	Walker Run	34.81	45.83	387.76	49.82	88.56
All	Walker Walk	13.53	56.73	897.81	34.94	64.56
All	Cheetah Run	34.66	187.55	318.41	67.53	125.68
All	Jaco Reach	23.95	167.85	287.55	18.64	52.98
All	Cartpole Swingup	56.82	78.57	787.52	67.56	97.81
Total Average		36.11	100.61	585.19	45.51	83.02

As the number of trajectories increases, the return for AT also increases. In some cases, AT is able to attain the desired target return by the 2nd or 3rd trajectory, resulting in a higher return in the last 4th trajectory. On the other hand, when multiple trajectories are rolled out using DT, the results are poor. DT is unable to produce consistent or higher returns beyond the 1st trajectory.

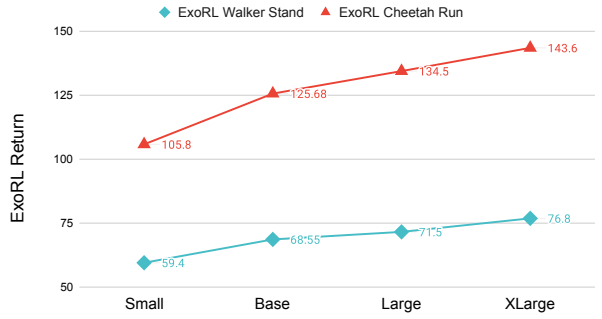

Figure 4. The results of Agentic Transformer with different model sizes on two ExoRL tasks.

4.4. Model Variations

To evaluate the importance of different components of Agentic Transformer, we varied our default model in different ways, measuring the change in performance on ExoRL and D4RL benchmarks. We present these results in Table 4.

In Table 4 rows (A), we vary the number of training trajectories n, keeping the number of testing trajectories constant. Performance improves with the increasing of number of training trajectories, this indicates a promising scaling direction for further improvement.

In Table 4 rows (B), we remove the task completion token 'd' from the input sequence, so the model is trained to 'blindly' learn from hindsight experience. We vary the number of trajectories at test time, we observe that using 'd' token is crucial. While without it Agentic Transformer still outperforms baselines, the performance degrades significantly compared with default configuration. In addition, without this completion token, the model does not improve with more trajectories at test time, indicating that completion token is important for the model to learn from hindsight experience.

In Table 4 rows (C), we observe that removing reward token 'r' has minimal negative effect. This is probably because the model can infer reward token by a simple subtraction from two consecutive returns-to-go tokens.

In Table 4 rows (D), we vary the desired return $\hat{R}_0$. Since default configuration uses 4 trajectories, the default target equals to total reward of last trajectory $\hat{R}_0 = \sum_{t=1}^{T} r_t^4$. We vary $\hat{R}_0$ to be the total reward of other trajectories. We observe that changing this target decreases performance significantly, with the largest decrease happens when $\hat{R}_0$ equals the total reward of the first trajectory.

In Table 4 rows (E), instead of having ordered trajectories $s = \left(\tau^1, \tau^2 \ldots, \tau^n\right)$, we randomly shuffle all τ for each training batch. We observed significantly worse results, in particularly on ExoRL, this change decreases the performance to only slightly better than BC and DT.

In Table 4 rows (F), we evaluate different number of trajectories at test time, we observed a steady better result from using more trajectories at test time. We further observe that although results are better with more trajectories, even using one trajectory, Agentic Transformer still outperforms Transformer-based policies on both ExoRL and D4RL benchmarks. This suggests that Agentic Transformer not only learns more than just imitation learning, but also learns to improve upon its own experience.

In Table 4 rows (G), we consider applying loss on all trajectories rather than just last trajectory. We observe that it is detrimental to performance, and particularly reduces perfor-

Table 4. Variations on the Agentic Transformer and chain of hindsight experience. Unlisted values are identical to those of the default configuration. All metrics are averaged over 3 random seeds based on the ExoRL and D4RL benchmarks.

Variants	With 'd'	With 'r'	Hindsight Tgt	Ordered	# Test Traj	# Train Traj	All tokens loss	ExoRL Avg	D4RL Avg
Default	true	true	4th	true	4	4	false	83.02	85.21
(A)						3		76.19	82.45
						2		65.47	80.85
						1		46.45	80.26
(B)	false				1			57.09	74.34
					2			61.92	73.56
					3			60.91	70.88
					4			61.20	75.68
(C)		false						76.59	80.43
(D)			1st					14.18	52.33
			2nd					32.29	65.55
			3rd					58.48	78.81
(E)				false				17.25	29.78
(F)					1			58.35	81.48
					2			74.17	82.56
					3			76.29	84.88
(G)					1		true	35.88	66.45
					2		true	66.30	71.55
					3		true	73.16	76.80
					4		true	73.88	78.88

mance for when the number of test trajectories is small. This suggests that it is best to optimize model towards 'better' behaviors rather than imitating all behaviors.

5. Related Work

5.1. Transformer for Decision-Making

Prior works explored using Transformers in the context of supervised or offline RL. Among them, decision transformer (DT) (Chen et al., 2021a) proposes to model trajectories as sequences and autoregressively predicts action conditioning on desired returns-to-go and past states and actions. Our model takes input as multiple trajectories and conditions on hindsight information for learning to improve. Chen et al. (2021a) found that DT does not benefit from longer context window and the results saturates at very short context length (*e.g.*, 3-5), possibly due to Markovian environments. Our Agentic Transformer (AT) models non-Markovian multiple episodes, it shows improved results with longer context length and benefits from Transformers architecture. Algorithm distillation (AD) (Laskin et al., 2022) also conditions the model on multiple trajectories, the difference is that AD requires the data to be the experience over the life time of a RL algorithm, while our model can learn from data from any sources. Another key difference is our model conditions on hindsight information including hindsight desired returns-to-go and hindsight task completion tokens. We observe these algorithm modifications are crucial for superior performance. Transformer has been explored in learning general world model (Liu et al., 2022a; Carroll et al., 2022; Wu et al., 2023), learning from multiple games (Reed et al., 2022; Lee et al., 2022), offline model-based learning (Janner et al., 2021; Liu et al., 2022a), meta learning (Melo, 2022; Team et al., 2023), vision-language navigation (Chen et al., 2021d; Shah et al., 2022), robot learning and behavior cloning from noisy demonstrations (Shafiullah et al., 2022; Cui et al., 2022), learning from multiple cameras (Seo et al., 2022), and language-conditioned imitation learning (Guhur et al., 2022; Liu et al., 2022b; Shridhar et al., 2022; Zheng et al., 2022). Since our model is a general decision-making model, applying it to these interesting tasks is possible.

5.2. Learning from Hindsight Experience

Learning from hindsight experience was explored in goal conditioned RL (Kaelbling, 1993; Andrychowicz et al., 2017; Schaul et al., 2015). Andrychowicz et al. (2017) proposes hindsight experience replay (HER) to relabel rewards and transitions retroactively to learn from sparse reward. In relation to HER (Andrychowicz et al., 2017), our work is in the batch setting rather than online setting. We propose algorithm improvement to construct hindsight experience directly from offline experience. HER is designed for Q-learning algorithms (Van Hasselt et al., 2016; Mnih et al., 2013; 2015) while AT use next token prediction to learn

from hindsight information. Chain-of-hindsight (Liu et al., 2023) explores turning all (binary or multi-scale) feedback into a sentence that consists of chain of all feedback and show improve improvements in aligning language models with human preferences. In relation to it, our work can be seen as applying chain-of-hindsight in the context of automatic feedback. Our work steers model's behavior using the desired target return and reward function at each step as feedback instead of using human preference.

5.3. Supervised and Meta RL

Motivated by transforming conventional RL (*e.g.*, policy gradient (Schulman et al., 2015; 2017) and Q-learning (Watkins, 1989; Mnih et al., 2013)) as a supervised learning problem, prior work explored various ways (Srivastava et al., 2019; Paster et al., 2020; Liu et al., 2022a; Carroll et al., 2022; Chen et al., 2021b; Laskin et al., 2022). Our work is closely related in that our model is similarly a return conditioned supervised learning. At test time, our model can self-improve based upon past experience to try to achieve target desired return. Using experience to improve model without changing weights is similar to few-shot or in-context learning in large language models (Brown et al., 2020). Recent work Algorithm Distillation (AD) (Laskin et al., 2022) demonstrates similar in-context behaviors in transformer model. AD is trained on the lifetime trajectories of a RL algorithm that can solves the task, posing a strong requirement of offline data, while in many important real world domains there exists only diverse, lower return data from multiple sources. In relation to AD, Agentic Transformer can be learned from sub-optimal data by turning the data into chain of hindsight experience. Leveraging online experience to improve model at test time is related to meta reinforcement learning (meta RL) (Duan et al., 2016; Wang et al., 2016). In meta RL the objective is to explicitly optimize for meta learning at test time, while Agentic Transformer does not, in contrast, the meta learning behavior emerges from training on chain of hindsight experience.

6. Conclusion

We propose Agentic Transformer (AT), a Transformer model with the ability of learning by directly combining information from multiple sub-optimal trials and being able to improve itself through multiple trials at test time. Motivated by prior works on hindsight experience replay and chain of hindsight, the key innovation behind Agentic Transformer is relabelling multiple trajectories to chain of hindsight experience that can be easily constructed from arbitrary offline data. On standard RL benchmarks, we showed AT outperforms both strong algorithms designed explicitly for offline RL as well as state-of-the-art Transformer-based policies.

Limitations and Future Work.

- *Large diverse datasets.* While Agentic Transformer (AT) outperforms prior transformer-based policies and performs competitively with TD-learning in standard RL benchmarks. AT is a GPT model therefore all limitations of transformer model still apply to AT. For instance, training AT requires large memory because of self-attention quadratic complexity and long sequence length. At test time, rollouting our model is sequential thus slower than non-transformer models. That being said, we believe the advantages of AT outweigh its drawbacks. As we observed in NLP and CV, it is worth scaling transformer-based policies in both model size and dataset size. As the datasets used in this work are still small, future work could explore scaling up dataset and model and have more investigation into using large transformer models for RL.

- *Real world applications.* As we observed in the experiments, Agentic Transformer can learn by directly combining information from multiple sub-optimal trials. Because diverse sub-optimal data is ubiquitous in the real world and AT scales well with model size and dataset diversity, we believe an interesting future direction is applying AT for real-world applications.

Acknowledgements

We thank the members of the Berkeley Robot Learning Lab and Berkeley AI Lab for helpful discussions, as well as Google TPU Research Cloud for granting us access to TPUs. This project is supported in part by ONR under N00014-21-1-2769.

References

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

Andrychowicz, M., Crow, D., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W. Hindsight experience replay. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5048–5058, 2017.

Andrychowicz, M., Raichuk, A., Stańczyk, P., Orsini, M., Girgin, S., Marinier, R., Hussenot, L., Geist, M., Pietquin, O., Michalski, M., Gelly, S., and Bachem, O. What matters in on-policy reinforcement learning? a large-scale empirical study. *arXiv preprint arXiv: Arxiv-2006.05990*, 2020.

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

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Burda, Y., Edwards, H., Storkey, A. J., and Klimov, O. Exploration by random network distillation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Carroll, M., Paradise, O., Lin, J., Georgescu, R., Sun, M., Bignell, D., Milani, S., Hofmann, K., Hausknecht, M., Dragan, A., et al. Unimask: Unified inference in sequential decision problems. *arXiv preprint arXiv:2211.10869*, 2022.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021a.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *CoRR*, abs/2106.01345, 2021b. URL `https://arxiv.org/abs/2106.01345`.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code. *arXiv preprint arXiv: Arxiv-2107.03374*, 2021c.

Chen, S., Guhur, P.-L., Schmid, C., and Laptev, I. History aware multimodal transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 34:5834–5847, 2021d.

Cui, Z. J., Wang, Y., Muhammad, N., Pinto, L., et al. From play to policy: Conditional behavior generation from uncurated robot data. *arXiv preprint arXiv:2210.10047*, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. Rl2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1582–1591. PMLR, 2018.

Guhur, P.-L., Chen, S., Garcia, R., Tapaswi, M., Laptev, I., and Schmid, C. Instruction-driven history-aware policies for robotic manipulations. *arXiv preprint arXiv:2209.04899*, 2022.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., van den Driessche, G., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J. W., Vinyals, O., and Sifre, L. Training compute-optimal large language models. *arXiv preprint arXiv: Arxiv-2203.15556*, 2022.

Janner, M., Li, Q., and Levine, S. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34: 1273–1286, 2021.

Kaelbling, L. P. Learning to achieve goals. In *IJCAI*, pp. 1094–1099. Citeseer, 1993.

Laskin, M., Yarats, D., Liu, H., Lee, K., Zhan, A., Lu, K., Cang, C., Pinto, L., and Abbeel, P. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

Laskin, M., Wang, L., Oh, J., Parisotto, E., Spencer, S., Steigerwald, R., Strouse, D., Hansen, S., Filos, A., Brooks, E., et al. In-context reinforcement learning with algorithm distillation. *arXiv preprint arXiv:2210.14215*, 2022.

Lee, K.-H., Nachum, O., Yang, M., Lee, L., Freeman, D., Xu, W., Guadarrama, S., Fischer, I., Jang, E., Michalewski, H., et al. Multi-game decision transformers. *arXiv preprint arXiv:2205.15241*, 2022.

Lee, L., Eysenbach, B., Parisotto, E., Xing, E., Levine, S., and Salakhutdinov, R. Efficient exploration via state marginal matching. *arXiv preprint arXiv:1906.05274*, 2019.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022.

Liu, F., Liu, H., Grover, A., and Abbeel, P. Masked autoencoding for scalable and generalizable decision making. *arXiv preprint arXiv:2211.12740*, 2022a.

Liu, H. and Abbeel, P. Aps: Active pretraining with successor features. In *International Conference on Machine Learning*, pp. 6736–6747. PMLR, 2021a.

Liu, H. and Abbeel, P. Behavior from the void: Unsupervised active pre-training. *arXiv preprint arXiv:2103.04551*, 2021b.

Liu, H., Lee, L., Lee, K., and Abbeel, P. Instruction-following agents with jointly pre-trained vision-language models. *arXiv preprint arXiv:2210.13431*, 2022b.

Liu, H., Sferrazza, C., and Abbeel, P. Chain of hindsight aligns language models with feedback. *arXiv preprint arXiv: Arxiv-2302.02676*, 2023.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Melo, L. C. Transformers are meta-reinforcement learners. In *International Conference on Machine Learning*, pp. 15340–15359. PMLR, 2022.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Paster, K., McIlraith, S. A., and Ba, J. Planning from pixels using inverse dynamics models. *arXiv preprint arXiv:2012.02419*, 2020.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2778–2787. PMLR, 2017.

Pathak, D., Gandhi, D., and Gupta, A. Self-supervised exploration via disagreement. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5062–5071. PMLR, 2019.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., Gimenez, M., Sulsky, Y., Kay, J., Springenberg, J. T., et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1312–1320. JMLR.org, 2015.

Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., and Moritz, P. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1889–1897. JMLR.org, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *Advances in Neural Information Processing Systems*, 2017.

Seo, Y., Hafner, D., Liu, H., Liu, F., James, S., Lee, K., and Abbeel, P. Masked world models for visual control. *arXiv preprint arXiv:2206.14244*, 2022.

Shafiullah, N. M. M., Cui, Z. J., Altanzaya, A., and Pinto, L. Behavior transformers: Cloning k modes with one stone. *arXiv preprint arXiv: Arxiv-2206.11251*, 2022.

Shah, D., Osinski, B., Ichter, B., and Levine, S. Lmnav: Robotic navigation with large pre-trained models of language, vision, and action. *arXiv preprint arXiv:2207.04429*, 2022.

Shridhar, M., Manuelli, L., and Fox, D. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pp. 894–906. PMLR, 2022.

Srivastava, R. K., Shyam, P., Mutz, F., Jaśkowski, W., and Schmidhuber, J. Training agents using upside-down reinforcement learning. *arXiv preprint arXiv:1912.02877*, 2019.

Team, A. A., Bauer, J., Baumli, K., Baveja, S., Behbahani, F., Bhoopchand, A., Bradley-Schmieg, N., Chang, M., Clay, N., Collister, A., et al. Human-timescale adaptation in an open-ended task space. *arXiv preprint arXiv:2301.07608*, 2023.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

Watkins, C. Learning from delayed rewards. 01 1989.

Wu, P., Majumdar, A., Stone, K., Lin, Y., Mordatch, I., Abbeel, P., and Rajeswaran, A. Masked trajectory models for prediction, representation, and control. *arXiv preprint arXiv:2305.02968*, 2023.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Reinforcement learning with prototypical representations. *arXiv preprint arXiv:2102.11271*, 2021.

Yarats, D., Brandfonbrener, D., Liu, H., Laskin, M., Abbeel, P., Lazaric, A., and Pinto, L. Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning. *arXiv preprint arXiv:2201.13425*, 2022.

Zheng, K., Chen, X., Jenkins, O. C., and Wang, X. Vlmbench: A compositional benchmark for vision-and-language manipulation. *Advances in Neural Information Processing Systems*, 35:665–678, 2022.

A. Experimental Details

The default length of chain of hindsight experience is four (*i.e.* input sequence consists of four trajectories) unless mentioned otherwise. For small and base model size, we distribute batch size 256 across multiple TPU devices and use gradient accumulation when necessary to reach effective batch size 256. For large and x-large model sizes, we distribute model weights across devices and similarly accumulate gradient to reach effective batch size 256. Our experiments are conducted on TPUv3 32 using Jax and Flax. On 32 TPUv3, each experiment takes around 4 hours on D4RL and around 6 hours on ExoRL. Models were trained for 10^5 gradient steps using the AdamW optimizer.

Our hyperparameters on all tasks are shown below in Table 5. In our preliminary experiments on ExoRL, we found that Agentic Transformer can condition on higher return targets, for fair comparison, we choose the return targets are chosen the same as in prior works. Specifically, on D4RL the target return equals to expert performance for each environment, except for 50% performance in HalfCheetah, and on ExoRL since the datasets are diverse and contain many lower return trajectories, we choose target returns based on TD3 performance.

Table 5. Hyperparameters of Agentic Transformers.

Hyperparameter	**Value**
Number of layers	3
Number of attention heads	1
Embedding dimension	128
Activation function	ReLU
Batch size	64
Dropout	0.1
Learning rate	10^{-4}
Learning rate decay	Linear warmup for 10^5 steps
Grad norm clip	0.25
Weight decay	10^{-4}
Initial desired target return at test time (D4RL)	6000 HalfCheetah 3600 Hopper 5000 Walker
Initial desired target return at test time (ExoRL)	850 Walker Stand 400 Walker Run 900 Walker Walk 350 Cheetah Run 300 Jaco Reach 800 Cartpole Swingup
Number of trajectories to form chain of hindsight experience during training	4
Number of trajectories at test time	4