T^3GDT: Three-Tier Tokens to Guide Decision Transformer for Offline Meta Reinforcement Learning

Zhe Wang, Haozhu Wang, Dmitriy Bespalov, Yanjun Qi
Amazon Web Services AI ML

Abstract

Offline meta-reinforcement learning (OMRL) aims to generalize an agent's knowledge from training tasks with offline data to a new unknown RL task with few demonstration trajectories. This paper proposes $\mathbf{T}^3$GDT: Three-tier tokens to Guide Decision Transformer for OMRL. First, our approach learns a global token from its demonstrations to summarize a RL task's transition dynamic and reward pattern. This global token specifies the task identity and prepends as the first token for prompting this task's RL roll-out. Second, for each time step t, we learn adaptive tokens retrieved from top-relevant experiences in the demonstration. These tokens are fused to improve action prediction at timestep t. Third, we replace lookup table-based time embedding with `Time2Vec` embedding that combines time neighboring relationships into better time representation for RL. Empirically, we compare $\mathbf{T}^3$GDT with prompt decision transformer variants and MACAW across five different RL environments from both MuJoCo control and MetaWorld benchmarks.

1 Introduction

Offline reinforcement learning [21] aims at approximating the optimal policy given a static dataset composed of the pre-collected interactions between the agent and the environment. Offline meta RL (OMRL) [18, 17] proceeds one step further, and the goal is to approximate the optimal policies for future unseen tasks when given only a few demonstrations from new tasks. During training, an OMRL agent will train on multiple tasks and learning-to-learn from a few trajectories collected for each task. During the evaluation, the OMRL agent faces new unseen tasks, and for each unseen task, the agent will condition on the demonstration to derive the task-specific policy from few-shot demonstrations. The derived policy then solves RL problems for that task by interacting with the environment.

Decision transformer (DT) [3] solves the offline RL by recasting the policy learning as a reward-conditioned sequence generation problem. The approach is gaining momentum for its simplicity and promising results. Inspired by the recent success of the few-shot generalization ability of the prompt-based language model framework, prompt decision transformer (PDT) [29] improves the DT via a prepended trajectory prompt formulation and uses the prompt sequence right before a test RL task' roll-out trajectory as the prefix of the input to the transformer.

In this paper, we propose a new method $\mathbf{T}^3$GDT: Three-Tier Tokens to Guide Decision Transformer for OMRL. The first component in the $\mathbf{T}^3$GDT is to learn a guiding token to specify a new task's identity from its demonstrations; we call this a *global token* . The second set of guiding tokens from the $\mathbf{T}^3$GDT is designed to help the action generation at a concrete timestep t. We name these tokens, *adaptive tokens* , and learn them by retrieving relevant experience from demonstrations for customized guidance. The third component is about a better *time token* representation. We revise DT's lookup table-based time embedding with `Time2Vec`, a new time embedding method. `Time2Vec`

NeurIPS 2023 Workshop on Robot Learning: Pretraining, Fine-Tuning, and Generalization with Large Scale Models, New Orleans, United States of America

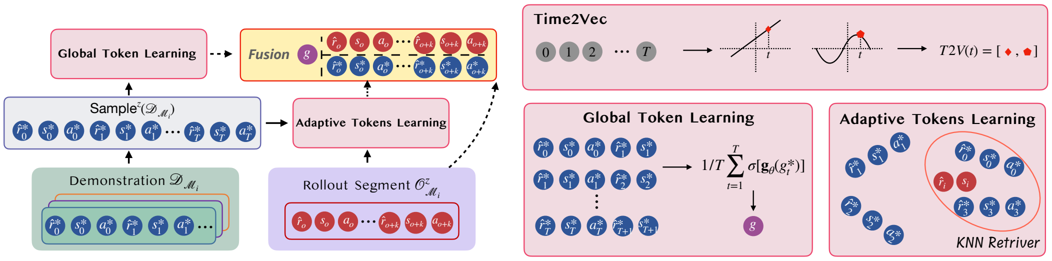

Global Token Learning
Fusion
g
$\hat{r}_o$ s_o a_o ⋯ $\hat{r}_{o+k}$ s_{o+k} a_{o+k}
$\hat{r}^*_o$ s^*_o a^*_o ⋯ $\hat{r}^*_{o+k}$ s^*_{o+k} a^*_{o+k}
Sample$^z(\mathscr{D}_{\mathscr{M}_i})$
$\hat{r}^*_0$ s^*_0 a^*_0 $\hat{r}^*_1$ s^*_1 a^*_1 ⋯ $\hat{r}^*_T$ s^*_T a^*_T
Adaptive Tokens Learning
Demonstration $\mathscr{D}_{\mathscr{M}_i}$
Rollout Segment $\mathscr{O}^z_{\mathscr{M}_i}$
Time2Vec
0 1 2 ⋯ T
t
t
$T2V(t) = [\,♦\,,\,⬟\,]$
Global Token Learning
$1/T \sum_{t=1}^{T} \sigma[\mathbf{g}_\theta(g^*_t)]$
Adaptive Tokens Learning
$\hat{r}_i$ s_i
KNN Retriver

Figure 1: The overall framework of the model $\mathbf{T}^3$GDT. The left panel summarizes the fusion process. The *global token* learns transition dynamic and reward function for task recognition; the *adaptive tokens* provide customized guidance for action generation at each t; the `Time2Vec` is light-weighted and encodes neighborhood timesteps into similar embedding vectors.

is more parameter efficient and encodes neighborhood timesteps into more similar embedding vectors. We use the *global token* as a prefix and summation-based embedding fusions for the *adaptive tokens* and *time token* to guide the decision transformer architecture for in-context learning-based few-shot policy generation in OMRL.

Overall, this paper makes the following contributions:

1. In $\mathbf{T}^3$GDT , the action generation will condition on learned three-tier tokens. Global token, learned from the transition dynamic and reward function, captures the task identity. Adaptive tokens, retrieved from the demonstrations, provide customized guidance for each action generation. Time tokens enable the agent to represent time better with fewer parameters.
2. Our empirical results demonstrate the competitive performance of $\mathbf{T}^3$GDT on environments from both MuJoCo control and the MetaWorld benchmarks. $\mathbf{T}^3$GDT outperforms SOTA baselines and is more effective than the full model fine-tuning baseline.

2 Method

2.1 Formulation

RL task can be formalized as a Markov decision process $\mathcal{M} := \langle S, A, \mathcal{R}, \mathcal{T}, \beta \rangle$, which consists of a state space S, an action space A, a reward function $\mathcal{R} : S \times A \to \mathbb{R}$, a transition dynamic $\mathcal{T} : S \times A \to S$, and an initial state distribution $s_0 \sim \beta$. A policy $\pi : S \to A$ will interact with the environment. At each timestep $t \geq 0$, an action $a_t \sim \pi(s_t)$ is output by the policy π and gets applied to the environment. After the agent performs action a_t, the environment transitions into the next state $s_{t+1} \sim \mathcal{T}(s_t, a_t)$ and produces a scalar reward $r_t \sim \mathcal{R}(s_t, a_t)$ as a feedback measuring the quality of the action a_t. During the evaluation phase, the optimality of the policy $\pi_{\mathcal{M}}$ is measured as the accumulated reward within a time horizon T [1]:

$$\pi^*_{\mathcal{M}} = \arg\max_{\pi} \sum_{t=0}^{T} r_t. \tag{1}$$

Instead of learning by interaction, in offline RL, the agent will learn from a static historical interaction trajectories, where each trajectory includes $\{s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_T, a_T, r_T\}$. OMRL targets at learning an agent that can approximate $\pi^*_{\mathcal{M}_j}$ for unseen tasks $\mathcal{M}_j$ given a handful of demonstrations by learning from multiple training tasks $\{\mathcal{M}_i\}_{i=1}^{n}$ [2]. To achieve positive cross-task knowledge transfer, OMRL assumes different tasks share the same state, same action space, and differ in their transition dynamics and the reward functions [31]. We denote training tasks as a set: $\{\mathcal{M}_i := \langle S, A, \mathcal{R}_{\mathcal{M}_i}, \mathcal{T}_{\mathcal{M}_i}, \beta_{\mathcal{M}_i} \rangle\}_{i=1}^{n}$. Similarly, we denote testing tasks as $\{\mathcal{M}_j := \langle S, A, \mathcal{R}_{\mathcal{M}_j}, \mathcal{T}_{\mathcal{M}_j}, \beta_{\mathcal{M}_j} \rangle\}_{j=1}^{n'}$. Every training task $\mathcal{M}_i$ (or test task $\mathcal{M}_j$) is associated with **a set of demonstration trajectory** denoted as $\mathcal{D}_{\mathcal{M}_i}$ (or $\mathcal{D}_{\mathcal{M}_j}$ for test task $\mathcal{M}_j$) that is composed of only a few, for instance, like 5 or 10, historical interaction trajectories from this RL task. The agent

[1] We focus on the environments with finite time horizon, but the definition generalizes to $T = \infty$. We also skip the constant discount factor for a better reading experience.

[2] Here we abuse the notations a little, using $\mathcal{M}$ denotes both an RL task and its MDP, for few math notations.

will learning-to-learn using training tasks' demonstration sets to derive task-specific policy $\pi_{\mathcal{M}_i}$ (or $\pi_{\mathcal{M}_j}$ during the evaluation). Training tasks $\{\mathcal{M}_i\}_{i=1}^n$ are associated with **a set of roll-out trajectories** denoted as $\{\mathcal{O}_{\mathcal{M}_i}\}_{i=1}^n$, on which we train their derived task-specific policies $\{\pi_{\mathcal{M}_i}\}_{i=1}^n$ to approximate $\{\pi^*_{\mathcal{M}_i}\}_{i=1}^n$. After training, the derived task-specific policy $\pi_{\mathcal{M}_j}$ is supposed to well-approximate $\pi^*_{\mathcal{M}_j}$.

Offline RL sequences have a unique property: for each t and task $\mathcal{M}_i$, adjacent tokens $\hat{r}_{i,t}, s_{i,t}, a_{i,t}$ are with different modalities, and the transition from timestep $t \to t+1$: $\hat{r}_{i,t}, s_{i,t}, a_{i,t} \to \hat{r}_{i,t+1}, s_{i,t+1}$, is fully determined by the transition dynamic $\mathcal{T}_{\mathcal{M}_i}$ and the reward function $\mathcal{R}_{\mathcal{M}_i}$, which capture the task identity of $\mathcal{M}_i$. Given the demonstration $\mathcal{D}_{\mathcal{M}_i}$, the agent should learn the task identity to guide the future action generation. This global token is coarse for action generation at each t. Adaptive guidance is desired for decision making at different timesteps. The agent should retrieve location-relevant experience from the demonstration set to imitate. Lastly, an intelligent agent is supposed to be time aware. Therefore, it can make customized decisions at different phases to handle complicated tasks. Given a sampled roll-out trajectory segment $\texttt{Segment}^z(\mathcal{O}_{\mathcal{M}_i})$ from $\mathcal{M}_i$:

$$[\hat{r}^z_{i,o},\ s^z_{i,o},\ a^z_{i,o}, \cdots, \hat{r}^z_{i,o+k},\ s^z_{i,o+k},\ a^z_{i,o+k}], \tag{2}$$

We introduce the global token and adaptive tokens learning from $\texttt{Segment}^z(\mathcal{D}_{\mathcal{M}_i})$, detailing the `Time2Vec` time embedding and knowledge fusion. We describe the overall framework in Figure 1.

Learning Global Token. When generating an RL sequence, the Markov transition dynamic and the reward function determine the transition across different timesteps. So we propose to learn the global token by summarizing the RL transition dynamic and reward pattern from timestep $t \to t+1$. The global token helps the agent to distinguish different tasks. Concretely, every data tuple $(\hat{r}^{z,*}_{i,t}, s^{z,*}_{i,t}, a^{z,*}_{i,t}, s^{z,*}_{i,t+1}, \hat{r}^{z,*}_{i,t+1})$ contains a screenshot of the $\mathcal{T}_{\mathcal{M}_i}$ and $\mathcal{R}_{\mathcal{M}_i}$. We concatenate the data tuple along the feature dimension as one data for the global token learning. Assume $\texttt{Segment}^z(\mathcal{D}_{\mathcal{M}_i})$ contains T such transition tuples, we apply the mean aggregator as set operator to learn the global token $g^z_{\mathcal{M}_i}$ to enjoy its permutation invariant property [32, 28]:

$$g^z_{\mathcal{M}_i} = \frac{1}{T}\sum_{t=0}^{T} \sigma(\mathbf{h}_{\theta_g}([\hat{r}^{z,*}_{i,t},\ s^{z,*}_{i,t},\ a^{z,*}_{i,t},\ s^{z,*}_{i,t+1},\ \hat{r}^{z,*}_{i,t+1}])), \tag{3}$$

where σ is the `GELU` activation, $\mathbf{h}_{\theta_g}$ is a linear layer with learnable parameters θ_g.

Same as all DTs, $\mathbf{T}^3$GDT uses a causal transformer [22] for auto-regressive sequence modeling. To guarantee the global token $g^z_{\mathcal{M}_i}$ will guide the action generation at all timsteps t, we prepend it right before $\texttt{Segment}^z(\mathcal{O}_{\mathcal{M}_i})$.

Learning Adaptive Tokens. The guidance from the global token $g^z_{\mathcal{M}_i}$ can be coarse when facing a specific action learning. The agent should condition on adaptive tokens. At each t, the action $a^z_{i,t}$ heavily depends on the current rtg $\hat{r}^z_{i,t}$ and the state $s^z_{i,t}$. We look back to the demonstration trajectory by retrieving the top-relevant experience. Concretely, we compare the similarity between $[\hat{r}^z_{i,t}, s^z_{i,t}]$ with those rtg-state pairs in $\texttt{Segment}^z(\mathcal{D}_{\mathcal{M}_i})$ and retrieve the top-m similar rtg-state-action tuples:

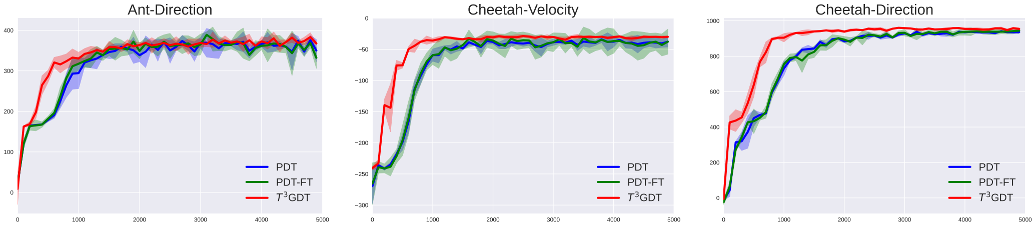
$$\{[\hat{r}^{z,*}_{i,t,m},\ s^{z,*}_{i,t,m},\ a^{z,*}_{i,t,m}]\} = \texttt{KNN}([\hat{r}^z_{i,t}, s^z_{i,t}] \leftrightarrows \texttt{Segment}^z(\mathcal{D}_{\mathcal{M}_i}),\ m),$$

where $\leftrightarrows$ represents the Euclidean distance comparison and the retrieval process. To summarize those top-m tuples, we use their mean as the final adaptive tokens at t.

$$[\hat{r}^{z,*}_{i,t},\ s^{z,*}_{i,t},\ a^{z,*}_{i,t}] = \frac{1}{m}\sum_{m} \mathbf{h}_{\theta_a}([\hat{r}^{z,*}_{i,t,m},\ s^{z,*}_{i,t,m},\ a^{z,*}_{i,t,m}]), \tag{4}$$

where $\mathbf{h}_{\theta_a}$ is a linear layer with learnable parameters θ_a.

Learning to Embed Time Tokens. An intelligent RL agent should represent time well. The lookup table-based time encoding is parameter heavy and independently encodes each t. The parameter size of this lookup table-based embedding layer grows linear with maximum length T. Also, this embedding does not consider the value and spatial relationship between time tokens when learning time representations. This is certainly less ideal. We propose to apply `Time2Vec` [12, 6] for the



(a) ANT-DIR (b) CHEETAH-VEL (c) CHEETAH-DIR

Figure 2: Meta-testing average performance of **T**^{3}GDT against baselines run over three random seeds facing unseen tasks. The x-axis represents the training epoch and y-axis is the average accumulated return on testing tasks.

time embedding to enable the agent to be time aware. `Time2Vec` projects a scalar time step t to an embedding vector of h dimension:

$$\texttt{T2V}(t)[i] = \begin{cases} \omega_i t/T + \varphi_i, & \text{if } \; i = 1. \\ \sin\left(\omega_i t/T + \varphi_i\right), & \text{if } \; 1 < i \leq h. \end{cases} \tag{5}$$

Here we need to learn parameter $\theta_t := \{\omega_i, \varphi_i\}$. `Time2Vec` contains a fixed number of parameters agnostic of the max timestep T. It can encode periodical events into the embedding. Moreover, adjacent timesteps have closer embeddings. Overall, `Time2Vec` is light-weighted, parameter efficient, and adjacency aware.

Before feeding into the causal transformer, the global token and the retrieval-enhanced rtg, states, action tokens will go through four different linear layers to project them to hidden spaces of the same dimensionality h as the time embedding. The projected tokens at each timestep will be added with their corresponding time embedding vector.

3 Results

We design experiments to demonstrate the few-shot ability of the **T**^{3}GDT on two RL benchmarks: MUJOCO control [26] and METAWORLD [31]. Three continuous control meta-environments of robotic locomotion are CHEETAH-DIR, CHEETAH-VEL, and ANT-DIR. Two other robotic arm manipulation environments, REACH and PICK&PLACE, are from METAWORLD, the results for REACHand PICK&PLACE are in the appendix. We have PDT, PTDT [8], and PDT-FT (PDT with full model finetuning) as baselines :

T^{3}GDT achieves consistent improvements over all baselines. **T**^{3}GDT achieves the best results compared with baselines on all five meta-environments. The major results are available in Figure 2 and Table 3. Table 3 shows that the two PDT variations including PDT-FT and PTDT show marginal improvements over PDT, and require either extra forward or backward passes for gradient estimation. On the other hand, **T**^{3}GDT gains significant improvements on CHEETAH-VEL, CHEETAH-DIR, ANT-DIR, and PICK&PLACE environments. The agent trained with **T**^{3}GDT largely surpassed the offline data collection policy $\bar{\mathcal{R}}(\mathcal{D}_{\mathcal{M}_j})$ on three out of five environments and achieved closer approximations on the other two: CHEETAH-VEL and REACH. Figure 2 shows that **T**^{3}GDT is training efficient with respect to the update, especially for the training epoch 0 → 1,000. **T**^{3}GDT quickly converges to better task-specific policies compared with other baselines. **T^{3}GDT outperforms full fine-tuning baseline.** While **T**^{3}GDT performs in-context learning, fine-tuning based approach performs in-parameter learning. It calculates/estimates the gradient on the data sampled from the few-shot demonstration. The agent trained with PDT already fits the demonstration set ideally. Therefore, extra updates bring marginal benefits. On the other hand, **T**^{3}GDT design enables stronger in-context learning. It is more efficient and stable.

4 Conclusion

Recasting the offline meta reinforcement learning (OMRL) task as a conditional sequence generation problem using transformers is promising. In this work, we propose a new model **T**^{3}GDT for OMRL. We first learn the global token $g^z_{\mathcal{M}_i}$ to encode the transition dynamic and the reward function, which specify new task' identities. Then local adaptive tokens are retrieved as the top-relevant experience from new task demonstrations. We also introduce the `Time2Vec` for better time embedding representations. The learned three-tier tokens guide the DT for action generations in new RL tasks' roll-out trajectories. Our method improves over SOTA baselines by providing DT with stronger in-context guidance learned from few shot demonstrations.

References

[1] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. B. Van Den Driessche, J.-B. Lespiau, B. Damoc, A. Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

[2] D. Brandfonbrener, A. Bietti, J. Buckman, R. Laroche, and J. Bruna. When does return-conditioned supervised learning work for offline reinforcement learning? *Advances in Neural Information Processing Systems*, 35:1542–1553, 2022.

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

[4] H. Furuta, Y. Matsuo, and S. S. Gu. Generalized decision transformer for offline hindsight information matching. *arXiv preprint arXiv:2111.10364*, 2021.

[5] A. Goyal, A. Friesen, A. Banino, T. Weber, N. R. Ke, A. P. Badia, A. Guez, M. Mirza, P. C. Humphreys, K. Konyushova, et al. Retrieval-augmented reinforcement learning. In *International Conference on Machine Learning*, pages 7740–7765. PMLR, 2022.

[6] J. Grigsby, Z. Wang, N. Nguyen, and Y. Qi. Long-range transformers for dynamic spatiotemporal forecasting. *arXiv preprint arXiv:2109.12218*, 2021.

[7] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020.

[8] S. Hu, L. Shen, Y. Zhang, and D. Tao. Prompt-tuning decision transformer with preference ranking. *arXiv preprint arXiv:2305.09648*, 2023.

[9] G. Izacard and E. Grave. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*, 2020.

[10] G. Izacard, P. Lewis, M. Lomeli, L. Hosseini, F. Petroni, T. Schick, J. Dwivedi-Yu, A. Joulin, S. Riedel, and E. Grave. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022.

[11] M. Janner, Q. Li, and S. Levine. Offline reinforcement learning as one big sequence modeling problem. In *Advances in Neural Information Processing Systems*, 2021.

[12] S. M. Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, S. Wu, C. Smyth, P. Poupart, and M. Brubaker. Time2vec: Learning a vector representation of time. *arXiv preprint arXiv:1907.05321*, 2019.

[13] U. Khandelwal, O. Levy, D. Jurafsky, L. Zettlemoyer, and M. Lewis. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*, 2019.

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

[15] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[16] J. Li, R. Jia, H. He, and P. Liang. Delete, retrieve, generate: A simple approach to sentiment and style transfer. In *North American Association for Computational Linguistics (NAACL)*, 2018.

[17] L. Li, R. Yang, and D. Luo. FOCAL: Efficient fully-offline meta-reinforcement learning via distance metric learning and behavior regularization. In *International Conference on Learning Representations*, 2021.

[18] E. Mitchell, R. Rafailov, X. B. Peng, S. Levine, and C. Finn. Offline meta-reinforcement learning with advantage weighting. In *International Conference on Machine Learning*, pages 7780–7791. PMLR, 2021.

[19] K. Paster, S. Mcilraith, and J. Ba. You cant count on luck: Why decision transformers and rvs fail in stochastic environments. *Advances in Neural Information Processing Systems*, 35:38966–38979, 2022.

[20] H. Peng, A. P. Parikh, M. Faruqui, B. Dhingra, and D. Das. Text generation with exemplar-based adaptive decoding. *arXiv preprint arXiv:1904.04428*, 2019.

[21] R. F. Prudencio, M. R. Maximo, and E. L. Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[22] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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

[24] S. Sodhani, A. Zhang, and J. Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pages 9767–9779. PMLR, 2021.

[25] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[26] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[27] K. Wang, H. Zhao, X. Luo, K. Ren, W. Zhang, and D. Li. Bootstrapped transformer for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:34748–34761, 2022.

[28] Z. Wang, J. Grigsby, A. Sekhon, and Y. Qi. ST-MAML: A stochastic-task based method for task-heterogeneous meta-learning. In *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022.

[29] M. Xu, Y. Shen, S. Zhang, Y. Lu, D. Zhao, J. Tenenbaum, and C. Gan. Prompting decision transformer for few-shot policy generalization. In *International Conference on Machine Learning*, pages 24631–24645. PMLR, 2022.

[30] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020.

[31] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

[32] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

[33] A. Zhang, S. Sodhani, K. Khetarpal, and J. Pineau. Learning robust state abstractions for hidden-parameter block mdps. *arXiv preprint arXiv:2007.07206*, 2020.

[34] M. Zhou, L. Yu, A. Singh, M. Wang, Z. Yu, and N. Zhang. Unsupervised vision-and-language pre-training via retrieval-based multi-granular alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16485–16494, 2022.

A Related Work

Offline RL as Sequence Generation. Treating policy learning in offline RL as a sequence generation problem via the language model is gaining momentum since DT [3]. Concurrent work is trajectory transformer [11]. TT discretizes independently every dimension of the state, action, and reward. It models both environment and the policy. During the evaluation, TT adapts beam search for planning. Bootstrapped Transformer [27] incorporates the idea of bootstrapping and leverages the learned model to self-generate more offline data to further boost the sequence model training. ESPER [19] analyzes that DT fails in the stochastic environment because the rtg term depends on environment stochasticity. It proposes to cluster trajectories and conditions the learning on average cluster returns. Brandfonbrener et.al [2] theoretically show that the successful scenarios for the return-conditioned decision transformer would require a stronger assumption on the sample complexity. Furuta et.al [4] suggests that DT is performing hindsight information matching and generalizing DT by replacing the rtg term with various other statistics of the future trajectory.

OMRL. Offline meta reinforcement learning (OMRL) targets approximating the task-specific optimal policy given a handful of static demonstrations from the task. MACAW [18] formalizes the OMRL setup and proposes to combine MAML with value-based RL. It increases the expressive power of the meta-learner by using the advantage regression as a subroutine in the inner loop. Most existing OMRLmethods are adapted from online meta-RL approaches, still rely on context-conditioned policy trained by TD-learning [25], which may lead to suboptimal performance.

Based on our knowledge, PDT [29] is the first work reframing the OMRL as a conditional sequence generation problem. It gains significant improvements by investigating the transformer architecture's strong ability to learn from a few examples and then generalize. Both $\mathbf{T}^3$GDT and PDT distill the policy in the offline dataset to the DT. AD [14] proposes to distill the RL algorithm to the DT by collecting a large enough offline dataset covering the learning history of the algorithm. Another loosely-related line of work is under the multi-task RL angle [23, 5, 4, 24, 30, 33]. However, the main target of the multi-task RL is learning one agent to handle all training tasks, instead of generalizing to future unseen tasks.

Retrieval-Enhanced Transformers. Retrieval enhanced transformer models are widely explored for various NLP tasks, we provide a short description on the related work in this line.

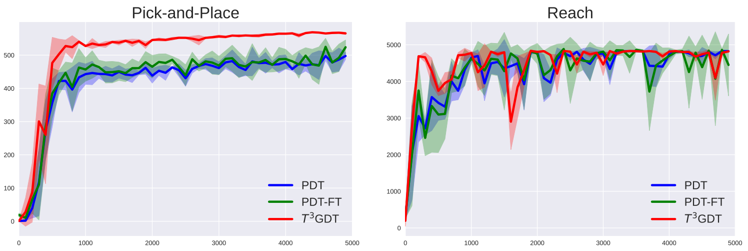
Rarely developed for RL, retrieval-enhanced transformers for NLP are well-explored. In the NLP domain, a small language model equipped with a retrieval module is capable of achieving on-par performance on various tasks compared with large language models [1, 13, 9]. The pretrained language models save the world knowledge in parameters and the retrievers capture the factual knowledge in a modular and interpretable paradigm. REALM [7] firstly proposes to jointly train end-to-end a retrieval system with an encoder language model for open-domain QA. Atlas [10] trained a retriever together with a seq2seq model and demonstrated its strong few-shot learning capabilities on various language tasks. It outperforms a 540B parameters model despite having 50x fewer parameters. RAG [15] designs finetuning approach for language models and neural retrievers for language generation. Besides, Peng et.al [20] retrieve exemplar text from training data as 'soft templates' for text summarization; Li et.al [16] design lexical-level similarity based retrieval for text style transfer; UVLP [34] propose retrieval-based multi-granular alignment for vision-and-language cross-modality alignment, etc.

B More Results

In this section, we first provide a summary of the environments we used, and then show the results on two other robotic arm manipulation environments, REACH and PICK&PLACE, from METAWORLD. See table 1 and figure. 3.

Table 1: A summary of the environments we used.

Env	$S\&A$-dim	# Training Tasks	# Test Tasks	Description	Variation
CHEETAH-VEL	20 & 6	35	5	A cheetah robot to run to achieve a target velocity	Target velocity
CHEETAH-DIR	20 & 6	2	2	A cheetah robot run to attain high velocity along forward or backward	Direction
ANT-DIR	27 & 8	45	4	A 8-joint ant agent to achieve high velocity along the specified direction	Goal Direction
REACH	39 & 4	15	5	A Sawyer robot to reach a target position in 3D space	Goal Position
PICK&PLACE	39 & 4	45	5	A Sawyer robot to pick and place a puck to a goal position	Puck and goal positions



(a) PICK&PLACE

(b) REACH

Figure 3: Meta-testing average performance of $\mathbf{T}^3$GDT against baselines run over three random seeds facing unseen tasks. The x-axis represents the training epoch and y-axis is the average accumulated return on testing tasks.

Table 2: Few-shot performance of the $\mathbf{T}^3$GDT for various environments. In the table, 'C-Vel', 'C-Dir', and 'A-Dir' represent the 'CHEETAH-VEL', 'CHEETAH-DIR', and 'ANT-DIR' environment respectively. We report the average and the standard deviation for three random seeds.

Models	C-Vel	C-Dir	A-Dir	Reach	Pick&Place
$\bar{\mathcal{R}}(\mathcal{D}_{\mathcal{M}_j})$	-23.5	900.4	351.47	4832.8	535.7
MACAW	-120.3 ± 38.6	500.8 ± 80.4	253.5 ± 3.8	3847.2 ± 74.4	450.8 ± 45.4
PDT	-37.9 ± 4.6	933.2 ± 11.4	375.6 ± 11.7	4827.2 ± 7.3	497.5 ± 34.8
PTDT	-39.5 ± 3.7	941.5 ± 3.2	378.9 ± 9.3	4830.5 ± 2.9	505.2 ± 3.7
PDT-FT	-40.1 ± 3.8	936.9 ± 4.8	373.2 ± 10.3	4828.3 ± 6.5	503.2 ± 3.9
$\mathbf{T}^3$GDT	$\mathbf{-26.7 \pm 2.3}$	$\mathbf{959.4 \pm 4.0}$	$\mathbf{383.3 \pm 10.4}$	$\mathbf{4832.2 \pm 5.2}$	$\mathbf{569.5 \pm 5.1}$

C Ablation Studies

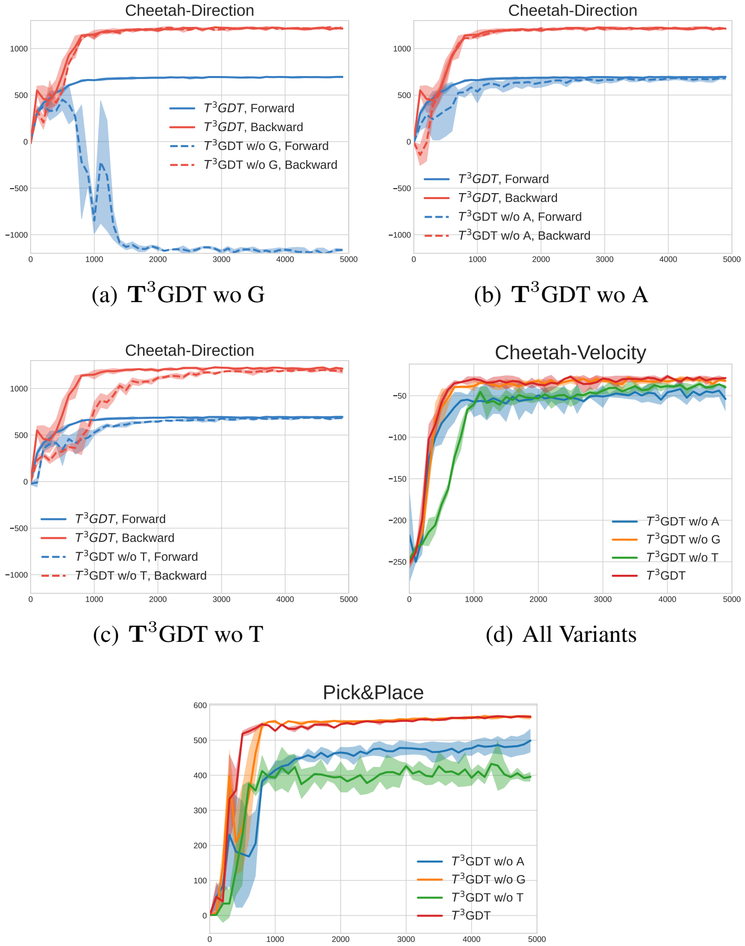
Ablation studies to show how learned tokens help. We learn three-tier tokens to guide decision transformer for OMRL. In this subsection, we empirically investigate how each tier of token helps with ablation studies. The ablation studies are designed to isolate each component and investigate their roles. Concretely, we have three variants: $\mathbf{T}^3$GDT wo G, which omits global token, $\mathbf{T}^3$GDT wo A, which omits adaptive tokens, and finally $\mathbf{T}^3$GDT wo T, in which we replace the proposed Time2Vec with the previously used lookup table. Table 4 (See Appendix) compares the results of all three variants and the full model on two robotic locomotion controls and one Sawyer robot control.

We design the global token $g^z_{\mathcal{M}_i}$ to learn from the transition dynamic $\mathcal{T}_{\mathcal{M}_i}$ and the reward function $\mathcal{R}_{\mathcal{M}_i}$, which are necessary and sufficient conditions for task distinguishment. Without the global token $g^z_{\mathcal{M}_i}$, the agent is confused with the task identity. For meta-learning environments where the test task identities differ significantly, the variant $\mathbf{T}^3$GDT wo G has drastically worse performance. For example, test tasks in CHEETAH-DIR include controlling a robot running to attain high velocity along either a forward or backward direction. The agent makes poor quality decisions if it fails at direction recognition. Figure 4(a) contains the accumulated rewards for both forward and backward test tasks. The agent trained with $\mathbf{T}^3$GDT wo G fails to recognize forward tasks, on which the accumulated reward is $< -1,000$, see the blue dashed curve at the bottom. To further investigate the role of the $g^z_{\mathcal{M}_i}$, we visualize their 2D projections in Figure 5 (See Appendix). Global tokens from different tasks $g^z_{\mathcal{M}_i}$ are well isolated and clustered from the same task.

In some cases, task identities are more similar to each other. For example for the CHEETAH-VEL, as described in Table 1, the variation is the target velocity, which is uniformly sampled from the range of 0 to 3. Similar task identities lead to closer task-specific policies. In this case, the role of the global token $g^z_{\mathcal{M}_i}$ will be downplayed, and the help from adaptive tokens will be dominant. Therefore, the performance of the variant $\mathbf{T}^3$GDT wo A, which removes the adaptive tokens, will be significantly impacted. See Figure 4(d) for the visualizations of all variants. Similar discussion goes to the PICK&PLACE environment, where the goal position of the objection is uniformly sampled within a square space, see the last column in Table 4 (See Appendix).

Table 3: Few-shot performance of the $\mathbf{T}^3$GDT for various environments. In the table, 'C-Vel', 'C-Dir', and 'A-Dir' represent the 'CHEETAH-VEL', 'CHEETAH-DIR', and 'ANT-DIR' environment respectively. We report the average and the standard deviation for three random seeds.

Models	C-Vel	C-Dir	A-Dir	Reach	Pick&Place
$\bar{\mathcal{R}}(\mathcal{D}_{\mathcal{M}_j})$	-23.5	900.4	351.47	4832.8	535.7
MACAW	-120.3 ± 38.6	500.8 ± 80.4	253.5 ± 3.8	3847.2 ± 74.4	450.8 ± 45.4
PDT	-37.9 ± 4.6	933.2 ± 11.4	375.6 ± 11.7	4827.2 ± 7.3	497.5 ± 34.8
PTDT	-39.5 ± 3.7	941.5 ± 3.2	378.9 ± 9.3	4830.5 ± 2.9	505.2 ± 3.7
PDT-FT	-40.1 ± 3.8	936.9 ± 4.8	373.2 ± 10.3	4828.3 ± 6.5	503.2 ± 3.9
$\mathbf{T}^3$GDT	$\mathbf{-26.7 \pm 2.3}$	$\mathbf{959.4 \pm 4.0}$	$\mathbf{383.3 \pm 10.4}$	$\mathbf{4832.2 \pm 5.2}$	$\mathbf{569.5 \pm 5.1}$



(a) $\mathbf{T}^3$GDT wo G

(b) $\mathbf{T}^3$GDT wo A

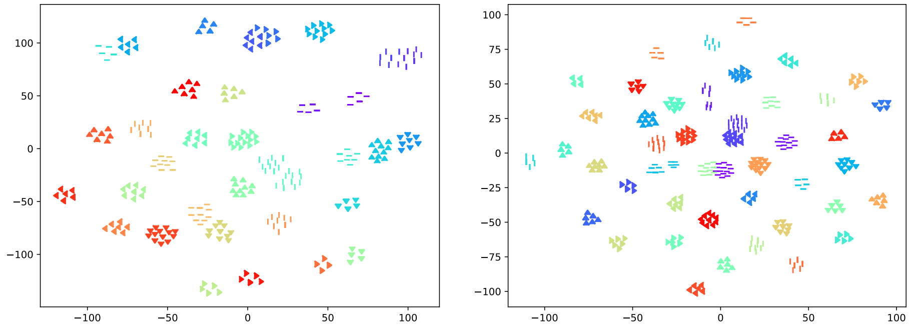
(c) $\mathbf{T}^3$GDT wo T

(d) All Variants

(e) All Variants

Figure 4: Ablation studies on CHEETAH-VEL CHEETAH-DIR, and PICK&PLACE. In (a)(b)(c), we compare each ablation with the full model on CHEETAH-DIR. Test tasks include running forward and backward. We show the accumulated reward for each task. The solid lines represent the full model $\mathbf{T}^3$GDT for both tasks. The dashed lines represent the result of each ablation version. For CHEETAH-DIR, the global token is more important. In (d) and (e), we show the results for CHEETAH-VEL and PICK&PLACE, where the adaptive tokens are more important. Curves represent the average accumulated reward on test tasks.

As in Figure 4(c) and 4(d), the introduced `Time2Vec` time embedding accelerates the convergence speed, especially at the beginning of the training phase. We attribute the advantage to the fewer



(a) C-Vel: 35 tasks.

(b) A-Dir: 45 tasks.

Figure 5: 2D projections of the global tokens $g^{z}_{\mathcal{M}_i}$.

parameters contained in the `Time2Vec`. The improvement brought by the `Time2Vec` surpasses both global and local tokens for PICK&PLACE.

Table 4: We design ablation studies by removing the global token $g^{z}_{\mathcal{M}_i}$, local tokens, and replacing the `Time2Vec` with lookup table for time embedding.

Models	C-Vel	C-Dir	Pick&Place
T^{3}GDT wo A	-47.8 ± 8.1	950.0 ± 9.7	499.0 ± 33.7
T^{3}GDT wo G	-33.0 ± 1.4	680.8 ± 106.9	568.0 ± 5.5
T^{3}GDT wo T	-31.3± 4.2	941.6± 4.1	432.5 ± 19.0
T^{3}GDT	-26.7 ± 2.3	959.4 ± 4.0	569.5 ± 5.1

Table 5: The robustness of **T**^{3}GDT for different hyperparameter combinations.

Env	$k' = 10$		$k' = 25$	
	$m = 3$	$m = 5$	$m = 3$	$m = 5$
A-Dir	369.7±16.7	380.0±2.4	374.7± 1.7	**390.5±5.3**
C-Dir	958.5±9.0	962.8±4.3	962.4± 7.0	**963.8 ± 3.1**
C-Vel	-28.1±4.5	-26.5±1.4	-27.4±1.7	**-25.7±1.5**

T^3**GDT is robust to hyperparameters.** Labeling the reward for each timestep within a full episode requires labor. It is desirable if we can still achieve OMRL goal with a few rtg-state-action tuples rather than full episodes as demonstrations. In this subsection, instead of conditioning on knowledge from a full demonstration trajectory for roll-out sequence generation, we show how **T**^{3}GDT performs when learning both the global and adaptive tokens from only a few rtg-state-action tuples. We note the number of tuples as k'. Moreover, when learning the adaptive tokens, we use `KNN` to retrieve the top-m similar rtg-state-action tuples and take their average as the adaptive tokens, see Eq.(4). Therefore, the retrieved size m is another important hyperparameter. We run **T**^{3}GDT with various combinations of the (m, k') and report both the mean and standard deviation across three random seeds.

Table 5 shows that **T**^{3}GDT is robust with respect to different hyperparameter combinations. **T**^{3}GDT is data efficient and does not have a strict requirement on the demonstration trajectories' length. It can achieve on-par performance by learning global and adaptive tokens from only a short demonstration trajectory whose length is $k' = 10$.