Multi-Game Decision Transformers

Anonymous Author(s)
Affiliation
Address
email

Abstract

A longstanding goal of the field of AI is a strategy for compiling diverse experience into a highly capable, generalist agent. In the subfields of vision and language, this was largely achieved by scaling up transformer-based models and training them on large, diverse datasets. Motivated by this progress, we investigate whether the same strategy can be used to produce generalist reinforcement learning agents. Specifically, we show that a single transformer-based model – with a single set of weights – trained purely offline can play a suite of up to 46 Atari games simultaneously at close-to-human performance. When trained and evaluated appropriately, we find that the same trends observed in language and vision hold, including scaling of performance with model size and rapid adaptation to new games via fine-tuning. We compare several approaches in this multi-game setting, such as online and offline RL methods and behavioral cloning, and find that our Multi-Game Decision Transformer models offer the best scalability and performance. We release the pre-trained models and code to encourage further research in this direction.

1 Introduction

Building large-scale generalist models that solve many tasks by training on massive task-agnostic datasets has emerged as a dominant approach in natural language processing [17, 11], computer vision [18, 5], and their intersection [55, 3]. These models can adapt to new tasks (such as translation [57, 70]), make use of unrelated data (such as using high-resource language to improve translations of low-resource languages [16]), or even incorporate new modalities by projecting images into language space [41, 67]. The success of these methods largely derives from a combination of scalable model architectures [69], an abundance of unlabeled task-agnostic data, and continuous improvements in high performance computing infrastructure. Crucially, scaling laws [34, 28] indicate that performance gains due to scale have not yet reached a saturation point.

In this work, we argue that a similar progression is possible in the field of reinforcement learning, and take initial steps toward scalable methods that produce highly capable generalist agents. In contrast to vision and language domains, reinforcement learning has seen advocacy for the use of smaller models [15, 44, 7] and is usually either used to solve single tasks, or multiple tasks within the same environment. Importantly, training across multiple environments – with very different dynamics, rewards, visuals, and agent embodiments – has been studied less significantly.

Specifically, we investigate whether a single model – with a single set of parameters – can be trained to act in multiple environments from large amounts of expert and non-expert experience. We consider training on a suite of 41 Atari games [8, 22] for their diversity, informally asking "Can models learn something universal from playing many video games?". To train this model, we use only the previously-collected trajectories from Agarwal et al. [1], but we evaluate our agent interactively. We are not striving for mastery or efficiency that game-specific agents can offer, as we believe we are still in early stages of this research agenda. Rather, we investigate whether the same trends observed in language and vision hold for large-scale generalist reinforcement learning agents.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

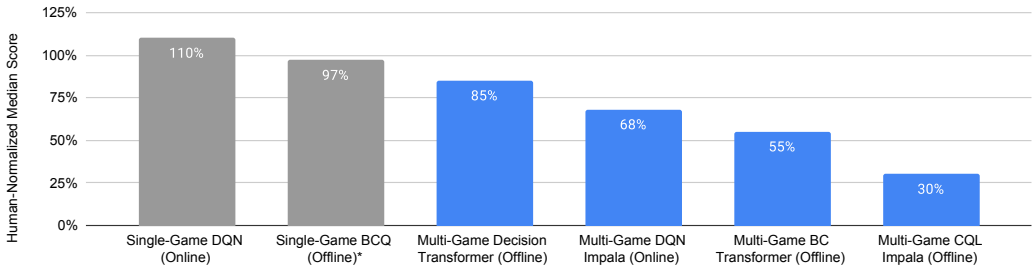


Figure 1: Aggregate human-normalized scores across 41 Atari games. Grey bars are single-game specialist models while blue are generalists. Single-game BCQ results are from Gulcehre et al. [22]. Multi-game models are all trained on a dataset [1] with median human-normalized score of 78%, which Multi-Game DT notably exceeds.

We find that we can train a single agent that averages 85% of human-level performance simultaneously across all games after training on offline expert and non-expert datasets (see Figure 1). Furthermore, we see similar trends that mirror those observed in language and vision: rapid fine-tuning to never-before-seen games with very little data (Section 4.5), a power-law relationship between performance and model size (Section 4.4), and faster training progress for larger models.

Notably, not all existing approaches to multi-environment training work well. We investigate several approaches, including treating the problem as offline decision transformer-based sequence modeling [13, 31], online RL [48], offline temporal difference methods [38], contrastive representations [51], and behavior cloning [54]. We find that decision transformer based models offer the best performance and scaling properties in the multi-environment regime. However, to permit training on both expert and non-expert trajectories, we find it is necessary to use a guided generation technique from language modeling to generate expert-level actions, which is an important departure from standard decision transformers.

Our contributions are threefold: First, we show that it is possible to train a single high-performing generalist agent to act across multiple environments from offline data alone. Second, we show that scaling trends observed in language and vision hold. And third, we compare multiple approaches for achieving this goal, finding that decision transformers combined with guided generation perform the best. It is our hope this study can inspire further research in generalist agents. To aid this, we plan to make our pre-trained models and code publicly available.

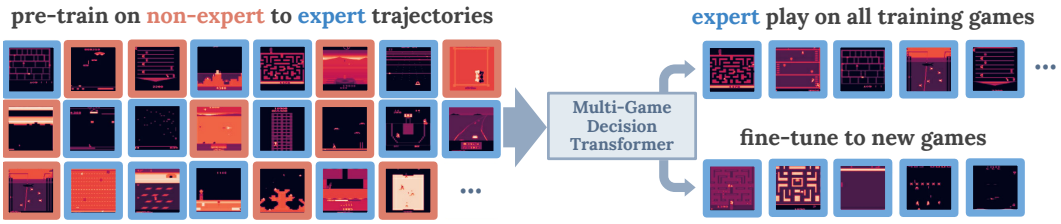


Figure 2: An overview of the training and evaluation setup. We observe expert-level game-play in the interactive setting after offline learning from trajectories ranging from beginner to expert.

2 Related Work

A generalist agent for solving a variety of environments has been a goal for artificial intelligence (AI) researchers since the inception of AI as a field of study [45]. This same reason motivated the introduction of the Atari suite (the Arcade Learning Environment, or ALE) as a testbed for learning algorithms [9]; in their own words, the ALE is for "empirically assessing agents designed for general competency." While the celebrated deep Q-learning [47] and actor critic [49] agents were among the first to use a single algorithm for all games, they nevertheless required separate training and hyperparameters for each game agent. Later works have demonstrated the ability to learn a

single neural network agent on multiple Atari games simultaneously, either online [19] or via policy distillation [53, 60]. The aim of our work is similar – to learn a single agent for playing multiple Atari games – with a focus on offline learning. We demonstrate results with human-level competency on up to 46 games, which is unseen in literature.

A closely related setting is learning to solve multiple tasks within the same or similar environments. For example in the robotics field, existing works propose to use language-conditioned tasks [43, 2, 30], while others posit goal-reaching as a way to learn general skills [46], among other proposals [33, 74]. In this work, we tackle the problem of learning to act in a large collection of environments with distinctively different dynamics, rewards, and agent embodiments. This complicated but important setting requires a different type of generalization that has been studied significantly less.

A concurrent work [58] also aims to train a transformer-based generalist agent based on offline data including for the ALE. This work differs from ours in that the offline training data is exclusively near-optimal and it requires prompting by expert trajectories at inference time. In contrast, we extend decision transformers [13] from the Upside-Down RL family [64, 61] to learn from a diverse dataset (expert and non-expert data), predict returns, and pick optimality-conditioned returns. Furthermore, we provide comparisons against existing behavioral cloning, online and offline RL methods, and contrastive representations [72, 51]. Other works that also consider LLM-like sequence modeling for a variety of single control tasks include [59, 76, 31, 20].

3 Method

We consider a decision-making agent that at every time t receives an observation of the world $\mathbf{o}^t$, chooses an action a^t, and receives a scalar reward r^t. Our goal is to learn a single optimal policy distribution $P^*_\theta(a^t|\mathbf{o}^{\leq t}, a^{<t}, r^{<t})$ with parameters θ that maximizes the agent's total future return $R^t = \sum_{k>t} r^k$ on all the environments we consider.

3.1 Reinforcement Learning as Sequence Modeling

Following [13], we pose the problem of offline reinforcement learning as a sequence modeling problem where we model the probability of the next sequence token x_i conditioned on all tokens prior to it: $P_\theta(x_i|x_{<i})$, similar to contemporary decoder-only sequence models [11, 14, 56]. The sequences we consider have the form:

$$x \;=\; \langle ..., \mathbf{o}^t_1, ..., \mathbf{o}^t_M, \hat{R}^t, a^t, r^t, ... \rangle$$

where t represents a time-step, M is the number of image patches per observation (which we further discuss in Section 3.2), and $\hat{R}^t$ is the agent's target return for the rest of the sequence. Such a sequence order respects the causal structure of the environment decision process. Figure 3 presents an overview of our model architecture.

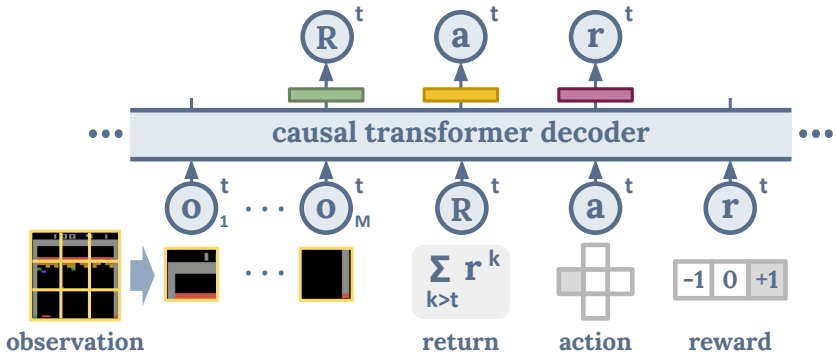


Figure 3: An overview of our decision transformer architecture.

Returns, actions, and rewards are tokenized (See Section 3.2 for details), and we train the model to predict the next return, action, and reward discrete token in a sequence via standard cross-entropy loss. The sequence we consider is different from Chen et al. [13], which has $\langle ..., \hat{R}^t, \mathbf{o}^t, a^t, ... \rangle$. Our

design allows predicting the return distribution and sampling from it, instead of relying on a user to manually select an expert-level return at inference time (See Section 3.4).

Predicting future value and rewards have been shown to be useful objectives for learning better representations in artificial reinforcement learning agents [42, 62, 40] and important signals for representation learning in humans [4]. Thus, while we may not directly use all of the predicted quantities, the task of predicting them encourages structure and representation learning of our environments. In this work, we do not attempt to predict future observations due to their non-discrete nature and the additional model capacity that would be required to generate images. However, building image-based forward prediction models of the environment has been shown to be a useful representation objective for RL [25, 24, 26]. We leave it for future investigation.

3.2 Tokenization

To generate returns, actions, and rewards via multinomial distributions similarly to language generation, we convert these quantities to discrete tokens. Actions a are already discrete quantities in the environments we consider. We convert scalar rewards to ternary quantities $\{-1, 0, +1\}$, and uniformly quantize returns into a discrete range shared by all our environments[1].

Inspired by simplicity and effectiveness of transformer architecture for processing images [18], we divide each observation image into a collection of M patches[2] (see Figure 3). Each patch is additively combined with a trainable position encoding and linearly projected into the input token embedding space. We experimented with using image tokenizations coming from a convolutional network, but did not find it to have a significant benefit and omitted it for simplicity.

We chose our tokenization scheme with simplicity in mind, but many other schemes are possible. While all our environments use a shared action space, varying action spaces when controlling different agent morphologies can still be tokenized using methods of [29, 39, 23]. And while we used uniform quantization to discretize continuous quantities, more sophisticated methods such as VQ-VAE [68] can be used to learn more effective discretizations.

3.3 Training Dataset

To train the model, we use an existing dataset of Atari trajectories (with quantized returns) introduced in [1]. The dataset contains trajectories collected from the training progress of a DQN agent [48]. Following [22], we select 46 games where DQN performance significantly exceeded that of a random agent. 41 games are used for training and 5 games are held out for out-of-distribution generalization experiments.

We chose 5 held-out games representing different game categories including `Alien` and `MsPacman` (maze based), `Pong` (ball tracking), `SpaceInvaders` (shoot vertically), and `StarGunner` (shoot horizontally), to ensure out-of-distribution generalization can be evaluated on different types of games.

For each of 41 games, we use data from 2 training runs, each containing roll-outs from 50 policy checkpoints, in turn each containing 1 million environment steps. This totals 4.1 billion steps. Using the tokenization scheme in previous sections, the dataset contains almost 160 billion tokens.

As the dataset contains agent's behavior at all stages of learning, it contains both expert and non-expert behaviors. We do not perform any special filtering, curation, or balancing of the dataset. The motivation to train on such data instead of expert-only behaviors is twofold: Firstly, sub-optimal behaviors are more diverse than optimal behaviors and may still be useful for learning representations of the environment and consequences of poor decisions. Secondly, it may be difficult to create a single binary criteria for optimality as it is typically a graded quantity. Thus, instead of assuming only task-relevant expert behaviors, we train our model on all available behaviors, yet generate expert behavior at inference time as described in the next section.

[1] The training datasets we use (Section 3.3) contains scalar reward values clipped to $[-1, 1]$. For return quantization, we use range $\{-20, ..., 100\}$ with bin size 1 in all our experiments as we find it covers most of the returns we observe in the datasets.

[2] We use 6x6 patches, where each patch corresponds to 14x14 pixels, in all our experiments.

3.4 Expert Action Inference

As described above, our training datasets contain a mix of expert and non-expert behaviors, thus directly generating actions from the model imitating the data is unlikely to consistently produce expert behavior (as we confirm in Section 4.7). Instead, we want to control action generation to consistently produce actions of highly-rewarding behavior. This mirrors the problem of discriminator-guided generation in language models, for which a variety of methods have been proposed [36, 71, 52].

We propose an inference-time method inspired by [36] and assume a binary classifier $P(\text{expert}^t|...)$ that identifies whether or not the behavior is expert-level before taking an action at time t. Following Bayes' rule, the distribution of expert-level returns at time t is then:

$$P(R^t|\text{expert}^t, ...) \propto P_\theta(R^t|...)P(\text{expert}^t|R^t, ...)$$

Similarly to [63, 65, 66, 35], we define a binary classifier to be proportional to future return with inverse temperature parameter κ:

$$P(\text{expert}^t|R^t, ...) \equiv \exp(\kappa R^t)$$

This results in a simple auto-regressive procedure where we first sample high-but-plausible target returns R^t according to log-probability $\log P_\theta(R^t|...) + \kappa R^t$, and then sample actions according to $P_\theta(a^t|R^t, ...)$. See Figure 4 for an illustration of this procedure and Appendix for implementation details. It can be seen as a variation of return-conditioned policies [37, 64, 13] that automatically generates expert-level (but likely) returns at every timestep, instead of manually fixing them for the duration of the episode.

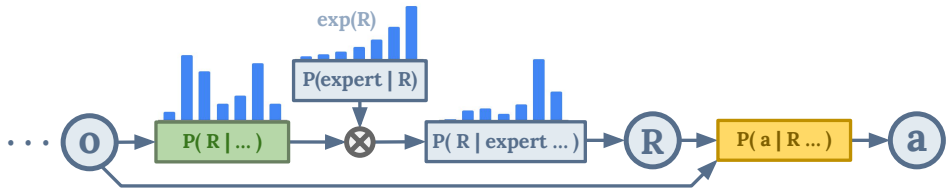


Figure 4: An illustration of our expert-level return and action sampling procedure. $P_\theta(R|...)$ and $P_\theta(a|R...)$ are the distributions learned by the sequence model.

Importantly, this formulation only affects the inference procedure of the model – training is entirely unaffected and can rely on standard next-token prediction frameworks and infrastructure. While we chose this formulation for its simplicity, controllable generation is an active area of study and we expect other more effective methods to be introduced in the future. As such, our contribution is to point out a connection between problems of controllable generation in language modeling and optimality conditioning in control.

4 Experiments

We formulate our experiments to answer a number of questions that are addressed in following sections:

- How do different **online and offline** methods perform in the multi-game regime?
- How do different methods **scale with model size**?
- How effective are different methods at **transfer to novel games**?
- Does **multi-game decision transformer** improve upon training data?
- Does **expert action inference** (Section 3.4) improve upon behavioral cloning?

We additionally qualitatively explore the behavior of these models, such as attention patterns and token embedding structure in the Appendix.

4.1 Setup

Model Variants and Scaling. We base our decision transformer (DT) configuration on GPT-2 [11] as summarized in the Appendix. We report DT-200M (a Multi-Game DT with 200M parameters) results if not specified otherwise. Other smaller variants are DT-40M and DT-10M. We set sequence length to 4 game frames for all experiments, which results in sequences of 156 tokens.

Training and Fine-tuning. We train all Multi-Game DT models on TPUv4 hardware and the Jaxline (Babuschkin et al. [6]) framework for 10M steps using the LAMB optimizer [73] with a $3 \cdot 10^{-4}$ learning rate, 4000 steps linear warm-up, no weight decay, gradient clip 1.0, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and batch size 2048. For fine-tuning on novel games, we train for 100k steps with a 10^{-4} learning rate and batch size of 256 instead.

Metrics. We measure performance on individual Atari games by human normalized scores (HNS) [48], i.e. $(\text{score} - \text{score}_{\text{random}})/(\text{score}_{\text{human}} - \text{score}_{\text{random}})$, or DQN-normalized scores, i.e. normalizing by the best DQN scores seen in the training dataset instead of using human scores. To create an aggregate comparison metric across all games, we use median human-normalized score, following Agarwal et al. [1].

4.2 Baseline Methods

BC Our Decision Transformer (Sec. 3.1) can be reduced to a transformer-based Behavioral Cloning (BC) [54] agent by removing the target return condition and return token prediction. Similar to what we do for Decision Transformer, we also learn BC models at different scales (10M, 40M, 200M parameters) while keeping other configurations unchanged.

C51 DQN As a point of comparison for online performance, we use the C51 algorithm [10] which is a variant of deep Q-learning (DQN) but with a categorical loss for minimizing the temporal difference (TD) errors. Following improvements suggested in Hessel et al. [27] as well as our own empirical observations, we use multi-step learning with $n = 4$. For the single-game experiments, we use the standard convolutional neural network (CNN) used in the implementation of C51 [12]. For the multi-game experiments, we modify the C51 implementation based on a hyperparameter search to use an Impala neural network architecture [19] with three blocks using 64, 128, and 128 channels respectively with a batch size of 128 and update period of 256.

CQL For an offline TD-based learning algorithm we use conservative Q-learning (CQL) [38]. Namely, we augment the categorical loss of C51 with a behavioral cloning loss minimizing $-\log \pi_Q(a|s)$, where (s, a) is a state-action pair sampled from the offline dataset and $\pi_Q(\cdot|s) = \mathrm{softmax}(Q(s, \cdot))$. Following the recommendations in Kumar et al. [38] we weight the contribution of the BC loss by 1 when using 100% of the offline data and 4 when using 1%. As in C51, we use a standard CNN network for single-game experiments and an Impala network for multi-game experiments. For scaling experiments, we vary the number of blocks and channels in each block of the Impala: the number of blocks and channels is one of $(5 \text{ blocks}, 128 \text{ channels}) \approx 5\text{M params}$, $(10 \text{ blocks}, 256 \text{ channels}) \approx 30\text{M params}$, $(5 \text{ blocks}, 512 \text{ channels}) \approx 60\text{M params}$, $(10 \text{ blocks}, 512 \text{ channels}) \approx 120\text{M params}$.

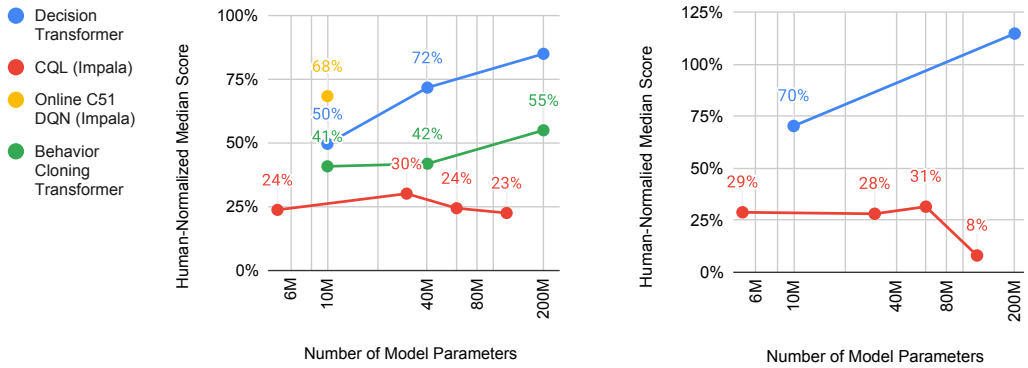
CPC, BERT, and ACL For rapid adaptation to new games via fine-tuning, we consider representation learning baselines including contrastive predictive coding (CPC) [51], BERT pretraining [17], and attentive contrastive learning (ACL) [72]. All state representation networks are implemented as additional multi-layer perceptrons (MLPs) or transformer layers on top of the Impala CNN used in C51 and CQL baselines. CPC uses two additional MLP layers with 512 units each interleaved with `ReLU` activation to represent $\phi(s)$, which is optimized by maximizing $\phi(s)^\top W \phi(s')$ of true transitions (s, s') and minimizing $\phi(s)^\top W \phi(\tilde{s})$ where $\tilde{s}$ is a state randomly sampled from the batch (including states from other games). For BERT pretraining, we use 2 self-attention layers with 4 attention heads of 256 units each and feed-forward dimension 512, and train $\phi(s)$ using BERT's masked self-prediction loss on a trajectory of sequence length 16. ACL shares the same model parametrization as BERT, with the inclusion of action prediction in the pretraining objective.

4.3 How do different online and offline methods perform in the multi-game regime?

We compare different online and offline algorithms in the multi-game regime and their single-game counterparts in Figure 1. We find that single-game specialists are still most performant. Among multi-game generalist models, our Multi-Game Decision Transformer model comes closest to specialist performance. Multi-game online RL with non-transformer models comes second, while we struggled to get good performance with offline non-transformer models. We note that our multi-game online C51 DQN median score of 68% compares similarly to multi-game median Impala score of 70%, which we calculated from results reported by [19] for our suite of games.

4.4 How do different methods scale with model size?

In large language and vision models, lowest-achievable training loss typically decreases predictably with increasing model size. Kaplan et al. [34] demonstrated an empirical power law relationship between the capacity of a language model (NLP terminology for a next-token autoregressive generative model) and its performance (negative log likelihood on held-out data). These trends were verified over many orders of magnitude of model size, ranging from few-million parameter models to hundreds of billion parameter models.



(a) Scaling of median scores for all training games with different model sizes and architectures.

(b) Scaling of median scores for all novel games after fine-tuning DT and CQL.

Figure 5: How model performance scales with model size, on training set games and novel games.

We investigate whether similar trends hold for *interactive* in-game performance – not just training loss – and show a similar power-law performance trend in Figure 5a. Multi-Game Decision Transformer performance reliably increases over two orders of magnitude, whereas the other methods either saturate, or have much slower performance growth.

We also find that larger models train faster, in the sense of reaching higher in-game performance after observing the same number of tokens. We discuss these results the Appendix.

4.5 How effective are different methods at transfer to novel games?

Pretraining for rapid adaptation to new games has not been explored widely on Atari games despite being a natural and well-motivated task due to its relevance to how humans transfer knowledge to new games. Nachum and Yang [50] employed pretraining on large offline data and fine-tunining on small expert data for Atari and compared to a set of state representation learning objectives based on bisimulation [21, 75], but their pretraining and fine-tuning use the same game. We are instead interested in the *transfer* ability of pretrained agents to new games.

We hence devise our own evaluation setup by pretraining DT, CQL, CPC, BERT, and ACL on the full datasets of the 41 training games with 50M steps each, and fine-tuning one model per held-out game using 1% (500k steps) from each game. The 1% fine-tuning data is uniformly sampled from the 50M step dataset without quality filtering. DT and CQL use the same objective for pretraining and fine-tuning, whereas CPC, BERT, and ACL each use their own pretraining objective and are

fine-tuned using the BC objective. All methods are fine-tuned for 100,000 steps, which is much shorter than training any agent from scratch. We additionally include training CQL from scratch on the 1% held-out data to highlight the benefit of rapid fine-tuning.

Fine-tuning performance on the held-out games is shown in Figure 6. Pretraining with the DT objective performs the best across all games. All methods with pretraining outperform training CQL from scratch, which verifies our hypothesis that pretraining on other games should indeed help with rapid learning of a new game. CPC and BERT underperform DT, suggesting that learning state representations alone is not sufficient for desirable transfer performance. While ACL adds an action prediction auxiliary loss to BERT, it showed little effect, suggesting that modeling the actions in the right way on the offline data is important for good transfer performance. Furthermore, we find that fine-tuning performance improves as the DT model becomes larger, while CQL fine-tuning performance is inconsistent with model size (see Figure 5b).

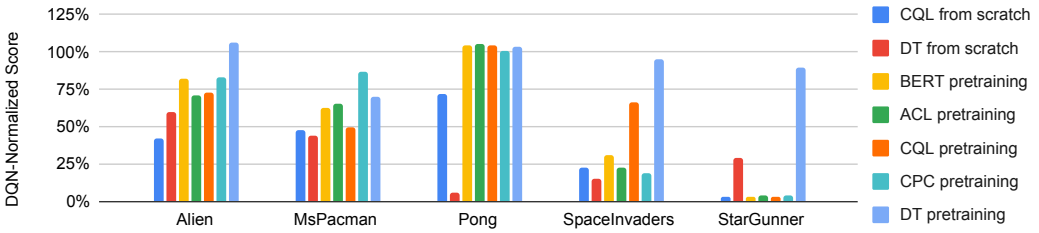


Figure 6: Fine-tuning performance on 1% of 5 held-out games' data after pretraining on other 41 games using DT, CQL, CPC, BERT, and ACL. All pretraining methods outperform training CQL from scratch on the 1% held-out data, highlighting the transfer benefit of pretraining on other games. DT performs the best among all methods considered.

4.6 Does multi-game decision transformer improve upon training data?

We want to evaluate whether decision transformer with expert action inference is capable of acting better than the best demonstrations seen during training. To do this, we look at the top 3 performing decision transformer model rollouts. We use top 3 rollouts instead of the mean across all rollouts to more fairly compare to the *best* demonstration, rather than an average expert demonstration. We show percentage improvement over best demonstration score for individual games in Figure 7. We see significant improvement over the training data in a number of games.

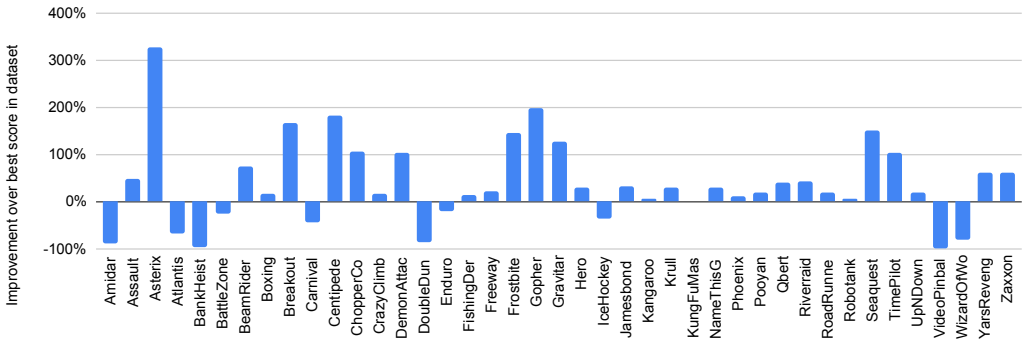


Figure 7: Percent of improvement of top 3 decision transformer rollouts over the best score in the training dataset. 0% indicates no improvement. Top-3 metric (instead of mean) is used to more fairly compare to the best – rather than expert average – demonstration score.

4.7 Does optimal action inference improve upon behavior cloning?

In Figure 1 we see that median performance across all games is indeed significantly improved by generating optimality-conditioned actions. Figure 8 shows the mean and standard deviation of scores

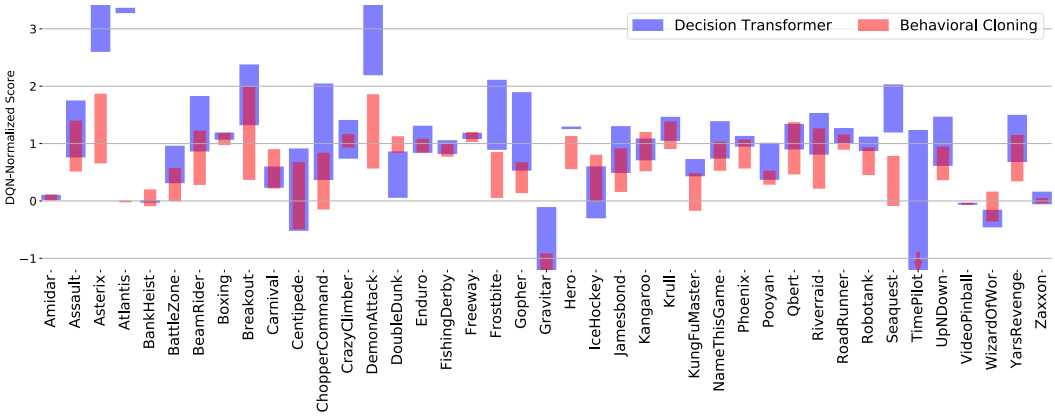


Figure 8: Comparison of per-game scores for decision transformer to behavioral cloning. Bars indicate $\pm$ standard deviation around the mean across 16 trials. We show DQN-normalized scores in this figure for better presentations.

across all games. While behavior cloning may sometimes produce highly-rewarding episodes, it is less likely to do so. We find decision transformer outperforms behavioral cloning in 31 out of 41 games.

5 Conclusion

In the quest to develop highly capable and generalist agents, we have made important and measurable progress. Namely, our results exhibit a clear benefit of using large transformer-based models in multi-game domains, and the general trends in these results – performance improvements with larger models and the ability to rapidly fine-tune to new tasks – mirror the successes observed for large-scale vision and language models. Our results also highlight difficulties of online RL algorithms in handling the complexity of multi-game training on Atari. It is interesting to note that our best results are achieved by decision transformers, which essentially learn via supervised learning on sequence data, compared to alternative approaches such as temporal difference learning (more typical in reinforcement learning), policy gradients, and contrastive representation learning. This begs the question of whether online learning algorithms can be modified to be as "data-absorbent" as DT-like methods. While even our best generalist agents at times fall short of performance achieved by agents trained on a single-task, this is broadly consistent with related works that have trained single models on many tasks [32, 58]. However, our best generalist agents are already capable of outperforming the data they are trained on. We believe the trends suggest clear paths for future work – that, with larger models and larger suites of tasks, performance is likely to scale up commensurately.

Limitations. We acknowledge reasons for caution in over-generalizing our conclusions. Our results are based largely on performance in the Atari suite, where action and observation spaces are aligned across different games. It is unclear whether offline RL datasets such as Atari are of sufficient scale and diversity that we would see similar performance scaling as observed in NLP and vision benchmarks. Whether we can observe other forms of generalization, such as zero-shot adaptation, as well as whether our conclusions hold for other settings, remains unclear.

Societal Impacts. In the current setting, we do not foresee significant societal impact as the models are limited to playing simple video games. We emphasize that our current agents are not intended to interact with humans or be used outside of self-contained game-playing domains. One should exercise increased caution if extending our algorithms and methods to such situations in order to ensure any safety and ethical concerns are appropriately addressed. At the same time, the capability of decision making based on reward feedback – rather than purely imitation of the data – has the potential to be easier to align with human values and goals.

References

[1] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pages 104–114. PMLR, 2020.

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[3] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

[4] William H Alexander and Samuel J Gershman. Representation learning with reward prediction errors. *arXiv preprint arXiv:2108.12402*, 2021.

[5] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6836–6846, 2021.

[6] Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Claudio Fantacci, Jonathan Godwin, Chris Jones, Tom Hennigan, Matteo Hessel, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Lena Martens, Vladimir Mikulik, Tamara Norman, John Quan, George Papamakarios, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Wojciech Stokowiec, and Fabio Viola. The DeepMind JAX Ecosystem, 2020. URL http://github.com/deepmind.

[7] Osbert Bastani, Yewen Pu, and Armando Solar-Lezama. Verifiable reinforcement learning via policy extraction. *Advances in neural information processing systems*, 31, 2018.

[8] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.

[9] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[10] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pages 449–458. PMLR, 2017.

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[12] Pablo Samuel Castro, Subhodeep Moitra, Carles Gelada, Saurabh Kumar, and Marc G Bellemare. Dopamine: A research framework for deep reinforcement learning. *arXiv preprint arXiv:1812.06110*, 2018.

[13] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34, 2021.

[14] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[15] Giuseppe Cuccu, Julian Togelius, and Philippe Cudré-Mauroux. Playing atari with six neurons. *arXiv preprint arXiv:1806.01363*, 2018.

[16] Rishabh Dabral, Ganesh Ramakrishnan, Preethi Jyothi, et al. Rudder: A cross lingual video and text retrieval dataset. *arXiv preprint arXiv:2103.05457*, 2021.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[19] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416. PMLR, 2018.

[20] Hiroki Furuta, Yutaka Matsuo, and Shixiang Shane Gu. Generalized decision transformer for offline hindsight information matching. *arXiv preprint arXiv:2111.10364*, 2021.

[21] Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning*, pages 2170–2179. PMLR, 2019.

[22] Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Thomas Paine, Sergio Gómez, Konrad Zolna, Rishabh Agarwal, Josh S Merel, Daniel J Mankowitz, Cosmin Paduraru, et al. Rl unplugged: A suite of benchmarks for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:7248–7259, 2020.

[23] Agrim Gupta, Silvio Savarese, Surya Ganguli, and Li Fei-Fei. Embodied intelligence via learning and evolution. *Nature communications*, 12(1):1–12, 2021.

[24] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2019.

[25] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[26] Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2020.

[27] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[28] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[29] Wenlong Huang, Igor Mordatch, and Deepak Pathak. One policy to control them all: Shared modular policies for agent-agnostic control. In *International Conference on Machine Learning*, pages 4455–4464. PMLR, 2020.

[30] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[31] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34, 2021.

[32] Lukasz Kaiser, Aidan N Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. One model to learn them all. *arXiv preprint arXiv:1706.05137*, 2017.

[33] Dmitry Kalashnikov, Jacob Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine, and Karol Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.

[34] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[35] Hilbert J Kappen, Vicenç Gómez, and Manfred Opper. Optimal control as a graphical model inference problem. *Machine learning*, 87(2):159–182, 2012.

[36] Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. Gedi: Generative discriminator guided sequence generation. *arXiv preprint arXiv:2009.06367*, 2020.

[37] Aviral Kumar, Xue Bin Peng, and Sergey Levine. Reward-conditioned policies. *arXiv preprint arXiv:1912.13465*, 2019.

[38] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

[39] Vitaly Kurin, Maximilian Igl, Tim Rocktäschel, Wendelin Boehmer, and Shimon Whiteson. My body is a cage: the role of morphology in graph-based incompatible control. *arXiv preprint arXiv:2010.01856*, 2020.

[40] Kuang-Huei Lee, Ian Fischer, Anthony Liu, Yijie Guo, Honglak Lee, John Canny, and Sergio Guadarrama. Predictive information accelerates learning in rl. *Advances in Neural Information Processing Systems*, 33: 11890–11901, 2020.

[41] Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*, 2021.

[42] Clare Lyle, Mark Rowland, Georg Ostrovski, and Will Dabney. On the effect of auxiliary tasks on representation dynamics. In *International Conference on Artificial Intelligence and Statistics*, pages 1–9. PMLR, 2021.

[43] Corey Lynch and Pierre Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020.

[44] Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

[45] John McCarthy, Marvin L Minsky, Nathaniel Rochester, and Claude E Shannon. A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. *AI magazine*, 27(4):12–12, 2006.

[46] Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and achieving goals via world models. *Advances in Neural Information Processing Systems*, 34, 2021.

[47] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[48] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[49] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[50] Ofir Nachum and Mengjiao Yang. Provable representation learning for imitation with contrastive fourier features. *Advances in Neural Information Processing Systems*, 34, 2021.

[51] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[52] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

[53] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*, 2015.

[54] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

[55] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[56] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[57] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[58] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent, 2022. URL https://arxiv.org/abs/2205.06175.

[59] Machel Reid, Yutaro Yamada, and Shixiang Shane Gu. Can wikipedia help offline reinforcement learning? *arXiv preprint arXiv:2201.12122*, 2022.

[60] Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

[61] Juergen Schmidhuber. Reinforcement learning upside down: Don't predict rewards–just map them to actions. *arXiv preprint arXiv:1912.02875*, 2019.

[62] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[63] Ross D Shachter. Probabilistic inference and influence diagrams. *Operations research*, 36(4):589–604, 1988.

[64] Rupesh Kumar Srivastava, Pranav Shyam, Filipe Mutz, Wojciech Jaśkowski, and Jürgen Schmidhuber. Training agents using upside-down reinforcement learning. *arXiv preprint arXiv:1912.02877*, 2019.

[65] Emanuel Todorov. Linearly-solvable markov decision problems. *Advances in neural information processing systems*, 19, 2006.

[66] Marc Toussaint. Robot trajectory optimization using approximate inference. In *Proceedings of the 26th annual international conference on machine learning*, pages 1049–1056, 2009.

[67] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34: 200–212, 2021.

[68] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6309–6318, 2017.

[69] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[70] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, 2021.

[71] Kevin Yang and Dan Klein. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*, 2021.

[72] Mengjiao Yang and Ofir Nachum. Representation matters: Offline pretraining for sequential decision making. In *International Conference on Machine Learning*, pages 11784–11794. PMLR, 2021.

[73] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[74] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020.

[75] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.

[76] Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. *arXiv preprint arXiv:2202.05607*, 2022.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 5
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 5
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code and models will be released.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]