Multi-Task Behavior Transformers

Urvi Shah, Yashvi Shah, Andrew Wu

***Abstract*—Offline RL algorithms are unable to work in domains lacking task-specific reward labels, while behavior cloning techniques are based on the premise that the data comes from a unimodal expert accomplishing a single task. In this work, we implement Behavior Transformers, a method for learning behaviors from multi-modal data. We predict multimodal actions by clustering continuous actions into discrete bins using k-means, and by utilizing the context-based multi-token prediction ability of transformer-based sequence models. We simultaneously develop a residual action corrector as we sample actions from BeT in order to provide continuous actions helpful for online rollouts. We apply BeT to the Franka Kitchen environment and Robomimic, enabling transfer learning, while witnessing performance boost and training efficiency due to transfer learning.**

I. Introduction

We present a multi-task version of Behavior Transformers that is used to model and learn behaviors from unlabeled multi-modal data. State-of-art RL algorithms often focus on reward-labeled data, which restricts their application to huge and pre-collected datasets and prevents their deployment in a setting with human-like demonstrations. BeT therefore seems to be a potential and intriguing approach to overcome this drawback. Another motivation is to use policies generated by BeT as a good prior for better downstream task solving akin to pretrained models and priors in natural language processing tasks.[34] We show that we can change the input and training paradigm to enable transfer learning and multi-task learning. Our results show significant training time speed ups with transfer learning but significant performance degradation for the multi-task setting.

II. Related Work

Demonstration-based offline learning: A Popular paradigm for training policies is imitation learning (IL) where policies are trained from a number of demonstrations. Offline imitation learning often entails variants of Behavioral Cloning [31], in which a policy is trained to produce the identical actions as the demonstration in each state. Robotic manipulation has made substantial use of offline imitation learning[17, 11, 1, 4, 38, 36, 42, 25, 41].IL often assumes the demonstration data to be optimal. In contrast, by utilizing reward annotations in the datasets, batch (offline) reinforcement learning[21], is a technique for learning from demonstrations that can include both good and bad quality data. BeT, a behavior cloning model, comes in the category of offline IL. Behavior cloning is a form of imitation l earning that aims to simulate the expert's behavior given the observation and is frequently applied in real-world settings[41, 42, 44, 13]. Behavior cloning techniques typically solve completely supervised learning problems, making them faster and easier to use than offline RL or reinforcement learning algorithms.

Behavioral learning with generative models: Inverse Reinforcement Learning, also known as IRL[33, 29]is an approach for imitation learning in which a model attempts to construct the reward function used to produce desired behavior. GAIL[16], an IRL algorithm, combines imitation learning and generative adversarial models to create a model that can generate behavior akin to an expert. Previous studies have made an effort to predict multi-modal, multi-human trajectories[20, 18] using this IRL paradigm. Gaussian Processes[32] have also been used in other works to develop dynamical models of human motion[39]. BeT's action factorization was inspired by another class of algorithms that learn a generative action decoder[30, 23, 35]from interaction data to speed up and facilitate downstream reinforcement learning. Finally, a group of algorithms, most notably[22, 28, 12, 19], train energy-based models rather than learning generative models directly. The desired behavior can then be produced by sampling these energy-based models.

Transformers for control: There has been a lot of interest in employing transformer models [37] to learn behavior and control because of their outstanding results in computer vision [10] and natural language processing[9, 2]. Among these, [5]applies transformer models to Reinforcement Learning and [18] applies it to Offline Reinforcement Learning while[6, 8, 24] use them for imitation learning. Transformers are mostly used by [24, 8] to sum up historical visual context, whereas [6] depends on their Long-term extrapolation capabilities to more effectively gather human-in-the-loop demonstrations. Both of these use cases served as inspiration for BeT, which use a transformer to condense historical context while utilizing its generative capabilities. Architecturally, BeT is most similar to the imitation learning variation of [18]; however, BeT learns the conditional distribution of action given state, whereas [18] learns the joint state, action distribution, allowing BeT to handle far more complex state spaces.

Distributionally multi-modal data sets: The gathering of behavior datasets that might help with downstream behavior learning has recently attracted attention. Some of these datasets [14, 23, 40] are open ended and lack reward or task labels. Many studies learning from multi-modal datasets attempt to learn a goal-conditioned model because the absence of labeled goal or reward labels implies that there is more multi-modality in the action distributions compared to action distributions of goal or reward conditioned datasets [14, 23, 15, 8]. The unlabelled datasets are also easier to collect because there are no labeling constraints, which should enable BeT to scale even

more in the future.

Robomimic: Robomimic[26] is a framework enabling robots to learn from demonstrations. It provides a large collection of demonstration datasets on robot manipulation domains as well as offline learning algorithms to learn from these datasets. In order to enable researchers and practitioners to fairly benchmark tasks and algorithms and to create the next generation of robot learning algorithms, robomimic makes robot learning widely accessible and reproducible. In this work we implement BeT on Lift PickPlaceCan, and NutAssemblySquare task.

III. Problem Formulation

Our problem is the classic behavioral cloning problem fitted to use transformers. We try to learn a policy π that maps a sequence of observations for some horizon h, $o_t, o_{t-1}, ...o_{t-h}$, to a sequence of actions to take given a data set that contains observations and actions, $D = (O, A)$, which in our case is multi-modal in that it captures multiple behaviors inside the dataset such as in Franka kitchen, placing the kettle versus opening a cabinet door.[34]

IV. Data

We utilize the robomimic [27] simulation framework for the simulation environment and data for the project. Robomimic is designed for assessing robot learning algorithms that learn by demonstration, having a wide variety of tasks, baselines, and collected human and machine demonstrations of various quality. Specifically, we train on the tasks of Lift, PickPlaceCan, and NutAssemblySquare that assess progressively harder tasks although of mostly overtly uni-modal nature since there is only a singular task to be accomplished in each task. Our experiments are trained on the Proficient-Human and Multi-Human dataset which includes 180 and 270 complete trajectories in the training dataset for each task stored in the hdf5 format of robomimic/robosuite respectively. The Multi-Human dataset consists of six operators as compared to the Proficient-Human dataset of only one so this dataset captures the multi-modal aspects of different ways of completing the same task that can occur in the wild. Currently, in our experiments we do not preprocess the data in any unique way besides the K-means clustering that Behavior Transformers require. In the future, action and observation encoders could be leveraged especially in the multi-task formulation.

A. Lift task details

The goal of the Lift task is to pick a cube off of a table with a robot arm. The simulation runs at 20 Hz with the action space consisting of a 7-dimensional vector for a single arm (translation of current end effector position, delta rotation of end effector, and opening/closing of gripper). The observation space is 10-dimensional with a 7-dimensional cube position and quaternion and a 3-dimensional cube position relative to the end effector. Cube z-pose is randomized and x,y inside a small center region of the table. [27]

B. PickPlaceCan

The goal of the Pick and Place Can task is to pick a can off the table and place it in the current bin. The observation space is slightly different, with 14-dimensions that include additionally the 4-dimensional can quaternion relative to the end effector. Can pose is randomized similarly to the lift task. [27]

C. NutAssemblySquare

The goal of the Nut Assembly Square task is to place a square nut through a square peg that exists on the table. The observation space is the same as Pick and Place Can with the 4-dimensional quaternion of the square nut to the end effector being exposed. The nut is randomized similarly to the lift task. [27]

V. Methods

A. BeT on Robomimic

Our objective is to learn a behavior policy $\pi : O \rightarrow A$ that accurately models the continuous observation and action pair data set containing the behaviors we are interested in, without the need of the reward labels or online interactions with the environments. This setup employs the Behavior Cloning concept, in which policies are learned to mimic expert rollouts demonstrations. In order to model multi-modal distributions of high-dimensional continuous variables, we employ continuous action binning with k-means algorithm that divides every action into 'action center' and a continuous 'offset'. With the aid of demonstration trajectories, we essentially train the transformer offline. For each ground truth action, we get a ground truth bin and residual action, which is utilized to train the minGPT with its binning and offset. In order to reconstruct a continuous action, we can combine the actions center and the corresponding residual action. After reconstructing the continuous action, it passes through an MLP network that attempts to reduce the losses between the continuous action we obtain from the k-means decoder and the actions of a specific batch. We have tried minimizing the total loss that comprises of smooth L1, L2, and cosine losses.

B. Transfer Learning

In transfer learning, we take a policy trained on one task and train it for another in a fine-tuning stage to try and achieve better performance than training from scratch in hopes of a "positive transfer" between tasks. Since the action dimension is the same between all three tasks we evaluate on, we need to pad the observation dimension between tasks in order to create a single model that can learn on all three tasks. Unlike MTL, we do not run K-means on the entire action space for all three tasks as that would leak information across tasks which is not in the nature of Transfer Learning. Instead, we keep a single K-means discretization of the action space.

Offset Loss Scale\Epochs	50	100	150	200	250
0.1	50	90	100	100	100
1	90	80	100	100	100
10	80	90	100	90	90
100	50	60	90	80	30
1000	70	70	80	60	60

TABLE I: Performance of BeT with respect to offset loss scale changes and number of epochs.

C. Multi-Task Learning

In multi-task learning, we take a policy and train it on multiple tasks at the same time. For us, this means training on Can, PickPlaceCan, and NutAssemblySquare all at the same time. In practice, we need to pad just as in transfer learning but also add a one-hot encoded vector to the observations that tells the model which task of the three it is training or evaluating currently. To extend the model architecture to multi-task, we run K-means on the entire action space of all three tasks to generate our clusters which is similar to task clustering MTL as discussed earlier. By creating a model that is general to all three tasks, we hoped to show that BeT could learn some common features between all three and improve success rates compared to BeT trained on single tasks.

VI. Experiments

A. Confirming BeT results

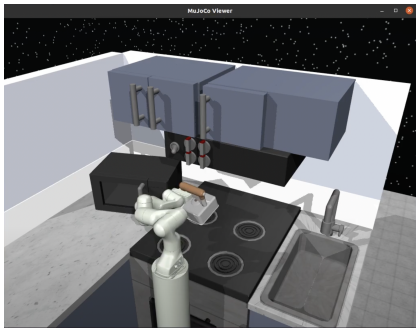
Using BeT's metrics in their own paper of percentage of successful roll-outs of number of tasks, we were able to confirm that our local model is close to the reported statistics. For the Franka Kitchen task, this was [100, 88, 75, 41, 2] successful completed tasks per 100 roll-outs versus the paper's [99, 93, 71, 44, 2]. A rollout of BeT implementation on Franka Kitchen environment is illustrated in Fig. 1.

B. Baselines

For our baselines we compare against the robomimic provided baselines across Lift, PickPlaceCan, and NutAssemblySquare.[27] Unlike the baselines, we validate our approach using only 20 rollouts for each task every 50 epochs and haven taken the best model out of a hyperparameter scan to use as our best, top-out model. Our best top-out single task model in Table 2 is trained for 250 epochs with similar meta-hyperparameters as BC-RNN such as window size and batch size.

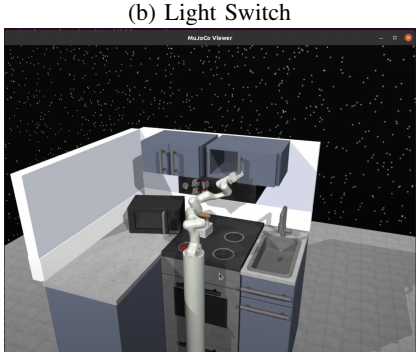
C. BeT on Robomimic

We trained BeT on Lift, PickPlaceCan, and NutAssemblySquare with the observation space as low-dim. Fig. 2 illustrates the implementation of BeT on robomimic datasets. We evaluated BeT using different number of bins, number of epochs, and offset loss scales. Table 1 shows that BeT performance improves when the offset loss scales are reduced, and the model is capable of achieving 100% accuracy in a limited number of epochs.

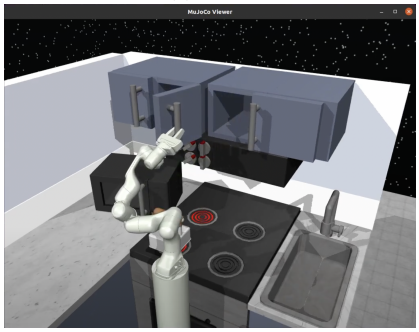


(a) Kettle

(b) Light Switch

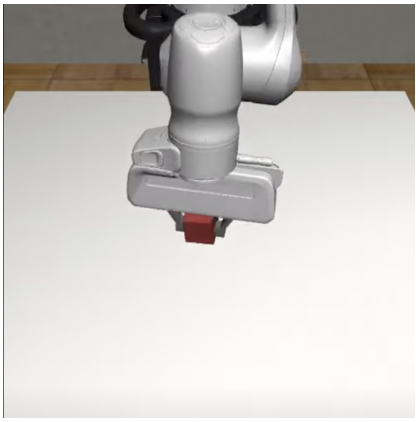


(c) Slide

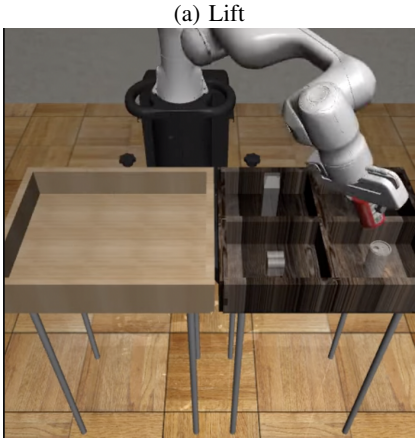


(d) Hinge

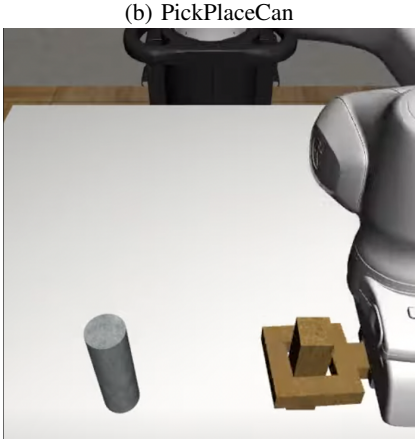
Fig. 1: Rollout of BeT on Franka Kitchen Environment



(a) Lift



(b) PickPlaceCan



(c) NutAssemblySquare

Fig. 2: BeT on Robomimic

D. Transfer Learning

Using the best top-out success rate model from the singular tasks, we evaluate the transfer between some permutations of the three tasks Lift, PickPlaceCan, and Square in Table 3. We train for 100 epochs on the first task and then train for 50 epochs on the next two tasks. Even though we train for 20 percent less in total than the single task top-out models and individually even lower than that, we find that transfer learning for BeT shows similar results for all tasks except Square. We find that in general, there is positive transfer from a harder task such as NutAssemblySquare to PickPlaceCan and Lift while there is negligible or slightly positive transfer from an easier task to a harder task such as Lift to Can or Square. Intuitively this makes sense and could be due to the K-means discretization capturing a richer action space for harder tasks than other tasks, although it seems that given enough training time on the individual task BeT can resolve this.

Dataset	BC	BC-RNN	BCQ	CQL	HBC	IRIS	BeT
Lift(PH)	100	100	100	92.7	100	100	100
Can(PH)	95.3	100	88.7	38	100	100	100
Square(PH)	78.7	84	50	5.3	82.6	78.7	55
Lift(MH)	100	100	100	56.7	100	100	100
Can(MH)	86	100	62.7	22	91.3	92.7	100
Square(MH)	52.7	78	14	.7	60.7	52.7	20

TABLE II: Best Top-Out Success Rates for Single Tasks[27]

Transfer Flow	Lift	PickPlaceCan	NutAssemblySquare
Can, Square, Lift	100	100	22
Square, Can, Lift	100	100	36
Lift, Can, Square	98	98	20

TABLE III: Best Top-Out Success Rates for Transfer Learning (PH)

E. Multi-Task Learning

In Multi-Task learning we run K-means on the action spaces of three tasks and we try to get BeT to learn three tasks inside a single model. Table 4 shows the different hyperparameters we needed to scan through to accommodate this change. We find that our best top-out model is larger (240 embeddings vs. 120), has twice as many bins (128 vs. 64), and needs to train for twice as long (500 vs. 250 epochs) as our best top-out model for single tasks and transfer learning. We find that in general training multi-task BeT does not provide competitive top-out success rates compared to the single or transfer learning models. We hypothesize that since the loss between all three tasks are split equally, it is substantially easier for the model to optimize for the easiest task, Lift, rather than learning the harder tasks since it tries to learn all the tasks at once. It is puzzling that the multi-task version is worse by a significant margin compared to transfer learning even though these three tasks share very similar primitive actions. Future work could explore warm starting the learning process by training on a single task first then switching to the full multi-task version after a specified amount of training time akin to transfer learning since we show that transfer learning has positive benefits to learning time.

F. Multi-Human vs. Proficient Human Datasets

We compare the multi-human dataset against the single, proficient-human dataset in robomimic to see if BeT can capture the multi-modal nature of multiple human operators and perform better than with the single proficient-human datase in Table 2. Although the multi-human dataset is larger by 100 trajectories than the proficient-human dataset, we find that there is significant degradation in performance of

Different Hyperparameters	Lift	PickPlaceCan	Square
Small Model (6 h, 6 l, 120 e)	100	70	0
Larger Model (6, 6, 240)	100	80	0
64 bins	100	55	0
128 bins	100	80	0
192 bins	100	70	0
256 bins	100	60	0
Mish Activation (500 epochs)	90	60	0
GELU Activation (500 epochs)	90	40	20

TABLE IV: Best Top-Out Success Rates for Multi-Task Learning (PH)

the NutAssemblySquare task for the multi-human dataset. We hypothesize that goal conditioning would be needed to counteract the effect of having lower quality operators exist inside the dataset.

VII. Conclusion

A. Summary

We implement Behavior Transformers (BeT), which employs a transformer with a discrete action mode predictor and a continuous action offset corrector to model continuous action sequences from multi-modal demonstrations. In order to model multi-modal distributions of high-dimensional continuous actions, we use the k-means approach to divide the continuous actions into discrete bins and residual action. In order to obtain better performance than training from scratch, we also employ transfer learning, where we take a policy taught on one task and train it for another in a fine-tuning step. We discover through transfer learning that training the more challenging activities first results in a large positive transfer (square to lift). Additionally, we implemented multi-task learning, but we realized that, when compared to single or transfer learning models, multi-task BeT training did not offer comparable top-out success rates.

B. Comparison to other class results

Sharachchandra's group in our class also tackles imitation learning with transformers. They utilize a stochastic policy modelled with GMMs and a transformer backend and attain similar single-task top-out results as us. This calls into question the necessity of the discretization process that the original BeT paper advocates for. Future work should look into the ability of Sharachchandra's architecture to capture multi-modal datasets such as MoMart and replacing the K-means discretization process of BeT entirely.

C. Future Work

On the dataset front, future work should extend the model to be able to work with more than just the three tasks of Lift, Can, and Square with a stronger focus on naturally multi-modal tasks such as the MoMart kitchen environment. For the model architecture, there is still the doubts about how necessary K-means clustering or any type of discretization on the action space is and should be researched further. Perhaps there is also better discretization methods such as VQ-VAEs or different clustering techniques like DB-SCAN, weighted K-means, agglomerative methods, etc. that should be tried. Action and observation encoders could be added to learn a better representation of the task's space. For meta learning, future work should try meta-learning frameworks and compare them against the multi-task formulation. In MTL, the feature learning approach should be tried if possible to implement it in tandem with discretization [43]. The original authors of BeT have already extended it with goal-conditioning and learning from play [7], however it is still in the single task formulation and goal-conditioned MTL would be interesting. Another idea is to utilize goal-conditioned BeT and generate massive amounts of play data such that the Transformer architecture can be scaled and show its true potential [3] and seeing if generalization can come naturally with more data such as it is in GPT since the 200-300 trajectories we work with is comparably small to the original BeT paper [34]. See BeT on Robomimic for the code release, built upon the foundations of BeT [34] and robomimic framework [27].

References

[1] Aude Billard, Sylvain Calinon, Ruediger Dillmann, and Stefan Schaal. Robot programming by demonstration. In *Springer handbook of robotics*, pages 1371–1394. Springer, 2008.

[2] 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.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL https://arxiv.org/abs/2005.14165.

[4] Sylvain Calinon, Florent D'halluin, Eric L Sauser, Darwin G Caldwell, and Aude G Billard. Learning and reproduction of gestures by imitation. *IEEE Robotics & Automation Magazine*, 17(2):44–54, 2010.

[5] 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*, 34:15084–15097, 2021.

[6] Henry M Clever, Ankur Handa, Hammad Mazhar, Kevin Parker, Omer Shapira, Qian Wan, Yashraj Narang, Iretiayo Akinola, Maya Cakmak, and Dieter Fox. Assistive

tele-op: Leveraging transformers to collect robotic task demonstrations. *arXiv preprint arXiv:2112.05129*, 2021.

[7] Zichen Jeff Cui, Yibin Wang, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. From play to policy: Conditional behavior generation from uncurated robot data, 2022. URL https://arxiv.org/abs/2210.10047.

[8] Sudeep Dasari and Abhinav Gupta. Transformers for one-shot visual imitation. In *Conference on Robot Learning*, pages 2071–2084. PMLR, 2021.

[9] 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.

[10] 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.

[11] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pages 357–368. PMLR, 2017.

[12] Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, pages 158–168. PMLR, 2022.

[13] Peter Florence, Lucas Manuelli, and Russ Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 5(2):492–499, 2019.

[14] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.

[15] Karol Hausman, Yevgen Chebotar, Stefan Schaal, Gaurav Sukhatme, and Joseph J Lim. Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets. *Advances in neural information processing systems*, 30, 2017.

[16] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

[17] Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 2, pages 1398–1403. IEEE, 2002.

[18] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.

[19] Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pages 5774–5783. PMLR, 2021.

[20] Namhoon Lee and Kris M Kitani. Predicting wide receiver trajectories in american football. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016.

[21] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[22] Minghuan Liu, Tairan He, Minkai Xu, and Weinan Zhang. Energy-based imitation learning. *arXiv preprint arXiv:2004.09395*, 2020.

[23] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. PMLR, 2020.

[24] Zhao Mandi, Fangchen Liu, Kimin Lee, and Pieter Abbeel. Towards more generalizable one-shot visual imitation learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2434–2444. IEEE, 2022.

[25] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Silvio Savarese, and Li Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. *arXiv preprint arXiv:2003.06085*, 2020.

[26] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[27] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation, 2021. URL https://arxiv.org/abs/2108.03298.

[28] Ofir Nachum and Mengjiao Yang. Provable representation learning for imitation with contrastive fourier features. *Advances in Neural Information Processing Systems*, 34:30100–30112, 2021.

[29] Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[30] Karl Pertsch, Youngwoon Lee, and Joseph Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on robot learning*, pages 188–204. PMLR, 2021.

[31] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

[32] Carl Edward Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (gpml) toolbox. *The Journal of Machine Learning Research*, 11:3011–3015, 2010.

[33] Stuart Russell. Learning agents for uncertain environments. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 101–103, 1998.

[34] Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone, 2022. URL https://arxiv.org/abs/2206.11251.

[35] Avi Singh, Huihan Liu, Gaoyue Zhou, Albert Yu, Nicholas Rhinehart, and Sergey Levine. Parrot: Data-driven behavioral priors for reinforcement learning. *arXiv preprint arXiv:2011.10024*, 2020.

[36] Albert Tung, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Learning multi-arm manipulation through collaborative teleoperation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9212–9219. IEEE, 2021.

[37] 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.

[38] Chen Wang, Rui Wang, Ajay Mandlekar, Li Fei-Fei, Silvio Savarese, and Danfei Xu. Generalization through hand-eye coordination: An action space for learning spatially-invariant visuomotor control. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8913–8920. IEEE, 2021.

[39] Jack M Wang, David J Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):283–298, 2007.

[40] Sarah Young, Jyothish Pari, Pieter Abbeel, and Lerrel Pinto. Playful interactions for representation learning. *arXiv preprint arXiv:2107.09046*, 2021.

[41] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021.

[42] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5628–5635. IEEE, 2018.

[43] Yu Zhang and Qiang Yang. A survey on multi-task learning. *CoRR*, abs/1707.08114, 2017. URL http://arxiv.org/abs/1707.08114.

[44] Yuke Zhu, Ziyu Wang, Josh Merel, Andrei Rusu, Tom Erez, Serkan Cabi, Saran Tunyasuvunakool, János Kramár, Raia Hadsell, Nando de Freitas, et al. Reinforcement and imitation learning for diverse visuomotor skills. *arXiv preprint arXiv:1802.09564*, 2018.