Extended Abstract

Motivation The recent papers in the field of Robotics and Machine Learned Motion Planning algorithms highlight the practical benefits of the action chunking. The known to the author papers focus on Imitation(supervised) Learning and do not cover the aspects of action chunking in application to the Reinforcement Learning. The goal of the course project is to document the practical aspects and the challenges of conducting the Reinforcement Learning with action chunks.

Method We research the problem of the Reinforcement Learning with action chunks on a particular task - bimanual insertion in the simulated ALOHA robot, environmentZipeng Fu (2023). This task has a moderate success rate with the imitation learning approach (oscillating around 60% on average with a high variance on different seeds and the large number of training steps). The course project aims to improve this success rate by fine-tuning the model using the feedback from the environment. The work is focused on improving the quality of the last predicted chunk as the imitation learning alone is able to master other parts of the trajectory.

Implementation Using a chunk of size K as a single action effectively shortens the trajectory K times and moves the complexity and variance in the learning from the level of the algorithm to the level of the neural network. It affects how the ideas from the traditional Reinforcement Learning could be applied to the task of fine tuning. With the reduced trajectories the true values of V, Q and Advantage could be directly calculated, however a very complex model becomes very fragile - wrong updates can break the behaviour learned from the demonstrations. In the course of the project two separate algorithms are implemented - the on-policy variation of the policy gradient method and off-policy actor-critic method, both of which are used in combination with the behaviour cloning on a limited set of original (scripted) demonstrations which provide the necessary stability to the learning process.

Results Evaluation of the algorithms shows that the first algorithm called "BC diluted by the policy gradient surrogate" improves the learning and success rate in comparison to the pure behaviour cloning by using the feedback from the environment on the same number of BC steps, however the behaviour cloning component plays a significant role in the learning and the highest evaluation success rate did not go above 80% showing degradation with further training, similarly to what happens to the Imitation Learning after reaching it's peak performance. The second algorithm "Actor-Critic method with chunk farming" reached a similar rate of 83.2% when ran in a conservative on-policy fashion similar to the first algorithm. The attempts to do a free off-policy learning showed overall worse results in a reasonable time range , however they showed that the model is able to learn a new behaviour. In the absence of cheap exploration, finding a working balance between the model stabilization and learning from the environment proved be the main challenge in the fine-tuning of the action chunks.

Discussion The both algorithms suffer from the lack of cheap exploration. Something like injecting a random noise during the training roll-outs does not seem practical, given the dimensionality of the action chunks. On practice, the exploration is effectively achieved by trying a different policy, which leads to undesired deviations in the policy and fragility of the training. The exploration-exploitation trade-off becomes a trade-off between doing more policy updates using the feedback from the environment and sticking to the agent's abilities obtained through the imitation learning. More policy updates driven by the environment feedback encourage the actor to visit more unseen combinations of states and actions but increase the risk of loosing the previous abilities learned by BC, the loss from which it takes long to recover. Given the above consideration some of the future topics that could be researched are: doing exploration using several actors or using a specific exploration algorithm tailored for the large chunks and the use of adaptive stabilization.

Conclusion The project achieves the goal of implementing and testing of the reinforcement learning algorithms for working with the models predicting action chunks. The results and the analysis are documented in the report. The project achieves the goal of learning the challenges of training the models using RL with action chunks and provides the observations and recommendation for any further research in this area. The report suggests some techniques that could be adapted to be used in the future work.

Deep Reinforcement Learning with Action Chunks. Fine-tuning ALOHA robot

Dmitry Usanov
SCPD/CGOE student
`dusanov@stanford.edu`

Abstract

The project researches the aspects of Reinforcement Learning with the models predicting sequence of actions executed by the agent in the open loop fashion. The problem is viewed in the context of fine-tuning ALOHA robot on a simulated task of bi-manual insertion. Several techniques are proposed. The challenges, achieved results, qualitative analysis and the suggestion for the future work in the field are presented in the report.

1 Introduction

The action chunking is a motion planning technique where the ML model predicts a series of actions that can be executed by the agent sequentially in the open loop fashion (as opposed to predicting a single action in the classic MDP framework). The idea of the action chunks was inspired by the studies in the neuroscience and it's application to the field of robotics can have the following practical benefits:

1. The learned behaviour with action chunks is able to reflect the temporal dependencies in motions such as the need to have pauses, which without action chunking would require a significantly more complex modelling.
2. The action chunking allows to reduce the frequency rate of the collected observations during the inference which could be an important factor for the performance of the systems with slower perception and can also free up the computational resources for different tasks.
3. Having a sequence of actions allows to perform a better smoothing of the machine learned robot trajectory.

The topic of action chunking appeared in the context of Imitation Learning, where it demonstrated high success rate on multiple tasks. The learning of action chunks from the environment, however, is not covered yet in the literature. On practice such need might arise in the context of fine-tuning. If after initial training of the model with action chunking with IL it is impossible or significantly more expensive to obtain more expert demonstrations for DAGGER or further BC training compared to collecting the data from the environment - reinforcement learning might become a solution.

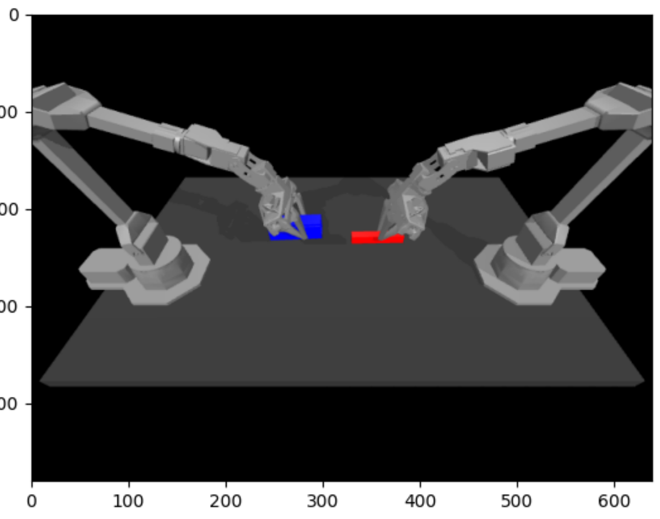
2 Related Work

The main paper in the field of action chunking for the robotics motion is **Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware**Tony Z. Zhao Vikash Kumar (2023) The paper describes a low cost system consisting of customized 6-Dof robotic arms ViperX , a smaller robot used for teleoperation (WindowX) which controls the main robot in the joint-space and 4 web-cameras streaming 480x640 RGB images - two on the main robot wrists, one camera at the top, and one at the front. The system is called ALOHA and it's used by the authors to demonstrate the performance of a newly proposed Imitation Learning algorithm called Action Chunking with Transformers (ACT). It is a policy implemented using the Transformer neural network architecture which gets the output of the convolutional neural networks for the camera data and the robot's joint positions as inputs and predicts a sequence of actions in the robot joint space. Despite predicting a series of consequent actions which could be executed in the open loop the authors propose to query the policy at each time step in order to collect several overlapping predictions which could be weight-averaged to achieve a smoother trajectory, this technique is called Temporal Ensemble. The tests demonstrate a significant improvement compared to the previous imitation learning algorithms in both the real world and simulated tasks. With a series of further experiments the authors show that the Action Chunking as the design concept plays a key role in improving the success rate.

The paperTony Z. Zhao Vikash Kumar (2023) does not cover the topic of reinforcement learning with ACT, the gap which this project aims to partially cover.

3 Experimental Setup

The course project uses Bimanual Insertion task and the imitation learning, in the simulated ALOHA environmentZipeng Fu (2023) as the baseline to research and document the aspects of fine-tuning of the models with action chunking using deep reinforcement learning.



Bimanual Insertion - the left and the right arms need to pick-up the socket and peg respectively, and then insert in mid-air so the peg touches the "pins" inside the socket.

On every step simulation provides the following rewards:

- 1: If both grippers touch the respective objects
- 2: Both grippers touch the respective objects, and the objects do not touch the table
- 3. Peg touches the socket and the both objects do not touch the table
- 4. Peg touches the pins inside the socket.
- 0. Otherwise

The experiment is successful when the highest reward in the trajectory is 4, that makes the task little different from the classic RL, instead of maximizing the sum of rewards we maximize the highest reward.

The simulated version is a mobile version of the Robot with only 3 cameras. The frameworkZipeng Fu (2023) provides the simulated environment, programs to generate the scripted expert demonstrations, and implementation of the ACT architecture. This course project re-uses the ACT code, some parts of the code for the neural network components for the critic implementation and the behaviour cloning part. The project uses it's own code for doing roll-outs, collecting the data from the environment, sampling, training and evaluating the reinforcement learning algorithms.

4 Implemented Algorithms

Two different algorithms were implemented in the course of the project. Both algorithms aim to improve only the last chunk (100 steps) of the robot trajectory. The decision was made based on observation that BC reaches the highest reward of 3 in more than 90% of the cases, but the final steps in the trajectory often do not lead to the reward of 4. For simplicity both algorithms aim to improve the highest reward from 3 to 4 and do not consider any trajectories resulted in the highest reward of 2 for the learning. (The decision that could be revisited in the future iterations as down the road the performance might significantly deviate and result of 2 might become more common). Both algorithms use the difference between the highest reward reached in the last chunk and 3.5 as either the Advantage of taking the action chunk (in the first algorithm) or the Q value for training the critic (in the second algorithm).

The first algorithm: BC diluted by the policy gradient surrogate

The target model is deterministic (which in general should be desired for robotics and high precision tasks), that means there is no standard notion of logarithm of probability of taking an action. There is some non determinism, however, in the certain layers of the neural network in training mode that results in a small non-zero MSE between the action from the trajectory and the action produced by the same policy on the same observation in the training mode. This fact is used to surrogate log_prob by a negative MSE, so the final objective function looks like:
$J(\theta) = -\frac{1}{N}\sum_{i=1}^{N}(MSE(a_i, \pi_\theta(s_i)) * A(s_i, a_i))$, where a_i is a chunk of actions from the trajectory,

The intuition behind this objective function is to incentivise the model to produce chunks which are similar to those that resulted in the positive advantage in a similar state and produce chunks which are different from those that resulted in the negative advantage in a similar state. A policy change resulted in the reaction to a negative sample is not expected to be precise, but when countered with the positive examples is expected to add some form of exploration to the algorithm which otherwise would have none. The inspiration for such behaviour is how humans do a similar thing in the real-time when try to insert one thing in another without looking, by trying to adjust the position by doing non-precise semi-random moves of the object.

The algorithm is only partially off-policy. The positive examples can be re-used, but the negative examples are expired after the policy is updated, thus the negative examples are prioritised in sampling of the mini-batches.

There is one gradient step per one policy roll-out. This results in the mini-batches consisting either fully from positive examples from the replay buffer or one new negative example countered by several positive examples. For stability the model is also trained with 50 gradient steps of the behaviour cloning (on the original demonstrations) after every 10 roll-outs - in that regard the algorithm can be viewed more as an assistance to the imitation learning using the experience from the environment, rather than an stand-alone learning algorithm.

The second algorithm: Actor-Critic method with "chunk farming"

In addition to the policy we also train the Critic network, which in case of ALOHA is a combination of Convolutional neural networks and a multi-layer perceptron which takes the observations and the action chunk as the input and produces a one dimensional output - Q value of the chunk. Using the critic network in computing the gradient of the actor policy addresses some of the problems of the first algorithm. The negative examples preserve their usefulness between the gradient steps and the updates computed by the gradient taken through the critic network are expected to be more specific than making a chunk just "different" from a wrong one.

The off-policy nature of the algorithm opens a possibility to greatly improve the data efficiency by leveraging the knowledge of the reward function of the specific task. In our example, each roll-out consists of 400 steps, and we want to fine-tune the last chunk of size 100, then based on the received rewards in each step in addition to the chunk produced at the step 300 we can use a sliding window with some randomization to produce, additional artificial chunks from a single episode and treat them as if they took place at the step 300. This new technique has the work name "chunk farming". In the last version of the algorithm it is implemented in the following way:

- For the positive episode, find the earliest step K with reward 4, randomly sample N chunks which start in the range [K-99, 300)
- For negative episode, in addition the the chunk starting on the step 300, take N chunks started in consequent steps.

The technique is designed to generate more data in the replay buffer with less roll-outs (doing roll-outs could be expensive in the real environment). In the conducted experiments N is 3.

For additional stability the algorithm uses a composite loss function. The positive samples in the mini-batch produce MSE loss with the predicted chunk and the loss on the negative samples in the mini-batch is computed using the negative of the Critic prediction with some weight γ. (In the conducted experiments γ is 0.005) The final loss of the mini batch is then computed as the mean across all samples in the batch. While it paves the way for the off-policy learning the algorithm increases the number of hyper paramters to tune. The algorithm has the hyper paramets for the number and the frequency of the gradient steps for training the critic and the actor, chunk farming rate, the weight for the critic-driven loss and has the freedom to sample the mini batches in various ways. The algorithm also uses regular BC steps (on the original set of demonstrations) for stabilization.

5 Results

The tests start with the model pre-trained on 10000 steps of the imitation learning. All success rates are computed on 250 roll-outs using one validation seed, different from the training seeds.

The results of the first algorithm.

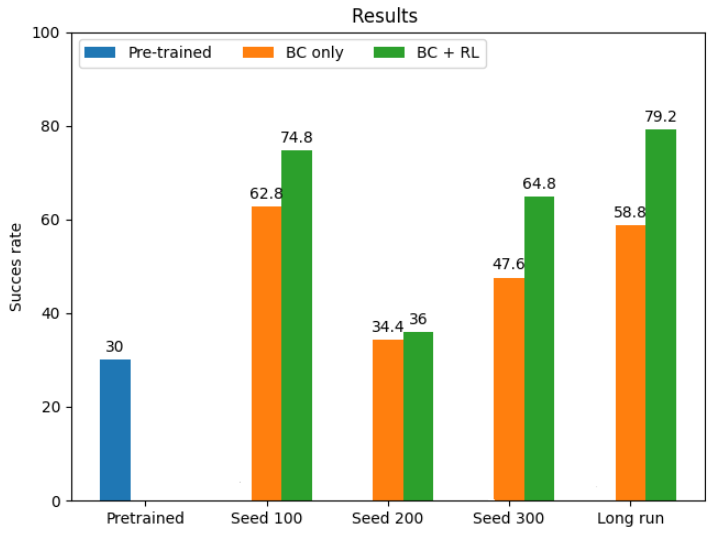


Figure 1: Visualization of results.

There are strong indications that the proposed algorithm enhances the learning of the model when the encoder option is enabled in ACT. The measuring method considers very high variance of both the learning and the validation and the high role of the behaviour cloning in the model training.

Test 1. Using three different training seeds, run the described algorithm on 500 roll-outs. For comparison run the version of the algorithms with the non-behaviour cloning part disabled on the same three seeds in a such way that behaviour cloning sees exact same batches on the same seeds in both versions. Validate the performance of the resulting 6 models. The result of the validation can be seen in Figure 1, in all three cases the additional gradient steps performed by the non-bc part of the algorithm improved the final performance of the model.

Test 2. The pre-trained model was trained using the proposed algorithm and it's BC-only version in a longer experiment of 5000 roll-outs. The results were validated, and the proposed algorithm performed significantly better than it's bc-only counterpart, the bc-only algorithm in fact showed degradation compared to a shorter run on the same seed (100). See figure 1.

Test 3. The continued training with even more 5000 roll-outs showed degradation of the performance, though RL version still performed better. Possible explanation - overfitting on the original demonstrations by he BC part of the algorithm.

Test 4. The method fails on the experiment with disabled encoder in ACT, the success rate degrades to single digits. Possible explanation - a high deviation in the training mode on the negative samples, which leads to larger gradient steps on the negative samples and destabilization.

The results of the second algorithm.

On a smaller number of steps the model showed mixed results in comparison to pure behavior cloning (in 2 of the 3 experiments AC performed worse), that can be attributed to a poor quality of the not warmed-up critic and a sub-optimal set of hyper parameters.

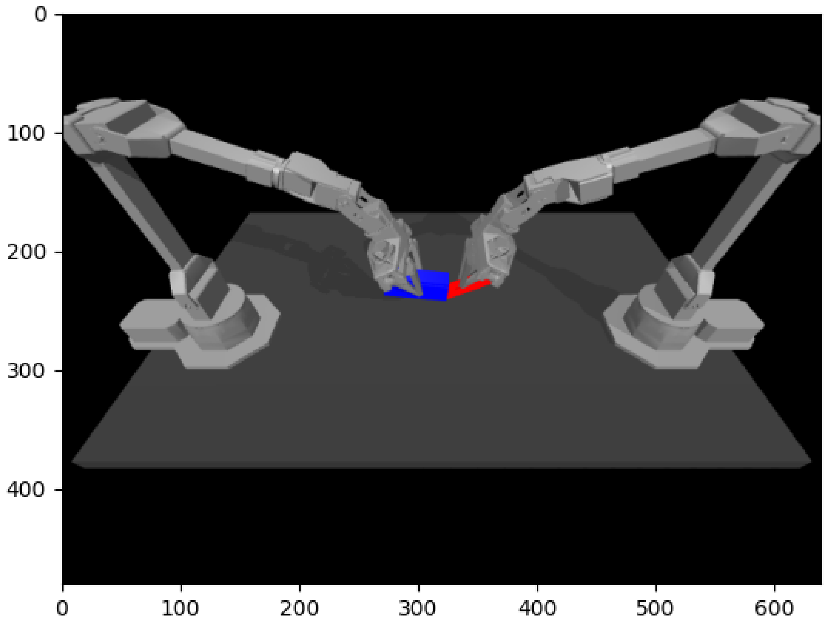
With the hyper-parameters set in a very conservative way similar to the first algorithm where the policy is updated once per roll-out with the mini-batches having only one last negative example, the algorithm achieves success rate of 83.2% after 8000 steps, with the evaluation error rate of the critic of only 5%. The model degrades with further steps in the same mode which can be attributed to the limits of the BC part of the training (it's possible over-fitting), as BC consitutes the largest part in that set up.

Attempts to do more RL gradient steps without the limitation descibed above from the begining of the training results in the model quickly loosing it's abilities obtained from the initial pretraining.

Attempts to improve the result by doing the free off-policy training after warming-up the critic and the actor on 8000 roll-outs resulted in a slow, oscilating but still declining accuracy of the model and the critic on a reasonable time range. The critic overestimation errors in this scenario indicate that more exploration (driven by the RL updates) could be required earlier in the training to avoid the decline. The fine-tuning of the hyper parameters and finding the optimal ratio of BC/RL steps is not trivial and with the accurate evaluation - it would require more time than is availbale to finalize the course project.

6 Qualitative analysis

BC only - qualitative analysis The pre-trained with BC model and the models trained further with only BC steps exhibited a characteristic failure mode, the peg (red) is being slightly lower than the socket and in the attempt to do the insertion the peg lower edge stuck which leads to a tilt. This failure corresponds to the majority of the cases when the BC-agent reaches reward 3 but not 4. The problem thath the proposed RL algorithms should solve.



Policy gradient - qualitative analysis

The model trained with the first policy gradient algorithm exhibited a more wide varuety of the failure modes, sometimes the opposite problem - the peg is positioned slightly higher than the socket and stuck by the upper edge - this varaity might indicate that the model recieves a fair share of exploration during the training.

Actor-critic - qualitative analysis

The actor-critic agent trained in a conservative on-policy way exhibits a noticeble share of the failures similar to the problem of the BC model - which indicates that this mode of training might not provide enough exploration.

The actor critic trained further in a free off-policy mode, however, demonstrates different problems and a new interesting behaviour - sometimes the arms stop, move back and then do the insertion, often successful. This makes an impression as if the robot learned to re-calibrate the position of the peg and the socket and indicates that the actor-critic method is able to train the policy with some new unseen behaviour. This might be a good sign for continuing training the model further with a larger replay buffer and larger batches to minimize the destabilization.

7 Derived observations and the ideas for the future work

Converting the predictive ability of the critic to the quality of the policy proved to be not fast. The main obstacle is understood as expensiveness of conducting the exploration which is done only by the changes to the current policy and subsequent roll-outs.

The observed pattern of overestimation errors by the critic on the negative evaluation examples, and underestimation errors on the positive evaluation examples in later experiments indicated that the critic learns to predict well the success of the policy it knows, but suffers from not knowing the outcomes on the unseen combinations of observations and action chunks.

The same lack of the cheap exploration also harms the first algorithm that could use more positive examples from the environment without risking to destabilize the model. A possible idea to address this problem could be using several actors branched from the main copy and updated on different batches which are used purely to collect the observations for training the critics and collecting the positive samples without the risk of harming the main copy of the actor. Alternatively we could come up with a more advanced way to conduct the exploration at the training time (the random noise does not look as a viable option given the dimensions of the chunk-actions).

The tuning of the hyper-parameters is constrained by the trade-off between the policy trying new actions and preserving the skills obtained from the imitation learning. In this process the evaluation error of the critic can be used as an indicator. For example, if the critic has the low error for predicting unsuccessful results it might indicate that the policy can benefit from more gradient steps driven by the reinforcement learning. The cases when the critic is over-optimistic and drives the policy in a wrong direction help correct the critic but cause a long turnaround for the policy to regain it's quality. Too many of such deviations could destabilize the policy. Some combination of the critic evaluation loss and the continuing policy evaluation loss could be used to dynamically adjust the hyper parameters at the training time to affect this trade-off.

Multiple critics with random initial weights, trained together and switched periodically could be used to increase the level of exploration if the future experiments achieve a flat success rate with no overestimation errors due to the lack of new exploration.

Finally, the large size of the training samples (specifically the images from the cameras) constitutes a big obstacle for accumulating a large replay buffer, the large models also limit the size of the mini batches (especially in the actor critic-method) - investing more in the workarounds is considered as an additional idea to improve the stability of the policy by increasing the diversity of the data seen in updates.

8 Conclusion

The course project achieves the goal of implementing several reinforcement learning algorithms for the application with the Action chunks. The algorithms were tried on practice and their performance, the practical challenges and the opportunities for improvements are described in this report. The time which is required to train and reliably evaluate the performance of the robot can be a constraining factor for the ability to conduct more experiments and try more ideas on practice - any further work should take this into account.

9 Team Contributions

- **Dmitry Usanov:** Learning the related works, setting up the simulation environment (fixing technical/compatibility issues of the simulation project), implementing the infrastructure for conducting the tests and recording the results, implementing a variation of the policy gradient algorithm, implementing the critic and a variation of an actor-critic method algorithm, implementing various data sampling strategies and conducting iterations on the parameters of the algorithms to achieve stability, conducting the tests, documenting the results, deriving the conclusions and proposing possible improvements.

Changes from Proposal The observation of the action chunk on practice made the author consider fine-tuning of the individual chunks as the most promising path forward, and the shifted complexity from the algorithm to the model itself made the author assume that a more simplistic architecture than ACT might be not up to the chosen task. (Also it's worth noting that the results with ACT are more valuable than the results on a more simplistic architecture for the usage in the real life application)

References

Chelsea Finn Tony Z. Zhao Vikash Kumar, Sergey Levine. 2023. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. (2023).

Tony Z. Zhao Zipeng Fu. 2023. https://github.com/MarkFzp/act-plus-plus. (2023).