A Fully Deep Learning Method for Language Conditioned Imitation Learning with Non-Atomic Actions

Yifan Zhou
Arizona State University
yzhou298@asu.edu

Neeraj Sreedhar
Arizona State University
nsreedh3@asu.edu

Nupoor Bibawe
Arizona State University
nbibawe@asu.edu

Anirudh Bhupalarao
Arizona State University
abhupal1@asu.edu

Abstract

Can robots follow human's natural language instructions and carry out corresponding actions? This question has been raising interest from both industry and academia for long. Recently, some work has been tackling this problem in the aspect of language conditioned imitation learning. However, all of these methods can only support short and atomic instructions, which contains only one action per instruction, and greatly reduces the possibility for robots to execute tasks of real-life-level complexity. In this paper, we propose a trial on tackling this problem, by separating the robot agent into two parts – a translator which analyzes human instructions and generate detailed sub-instructions, and an executor which carries out these detailed instructions. Experiments demonstrate a non-zero success rate on complex instructions, which is never observed on baselines.

1 Introduction and Related Work

Enabling robots to follow human commands in the form of natural languages has been a grand hope for many researchers (1). However, this problem is extremly difficult, since it usually requires the robot system to understand and make use of high dimensional modalities such as human natural language, computer vision, robot controlling signals and so on, but even only one of these modalities has been complex enough and is not a fully solved problem. Combining all of the modalities are exponentially more difficult, since the system needs to handle the combination of the modalities and have to inter-relate between them. For example, when the human tester says "Grab the apple", the robot needs to understand what an apple is, find an apple, and plan the controlling actions for moving towards the apple and grabbing it up.

Recently, with the fast development of machine learning and especially deep learning, there emerged methods to understand and make use modalities such as language (2; 3; 4) or vision (5; 6), and upon both of them (7; 8), some attempts of controlling robots' manipulation by human languages have observed initial successes (9; 10; 11; 12; 13). By leveraging FiLM-style (14) modality fusion networks, BC-Z (13) uses behavior cloning to train a joint language, vision and control model by introducing a large dataset with 100 tasks. ModAttn (15), on the other hand, leverages attention mechanism to form modularity in this imitation learning process, and achieves success in such language conditioned manipulations tasks, and is able to transfer the trained policy to new robots and environments in a data-efficient manner. Although these successes are observed, they are only limited to short instructions which typically contain only one atomic action, such as pick and/or place, push, swipe and so on. It remains an unsolved problem to enable the agent to perform longer actions, due to 2 potential facts: Firstly, current imitation learning system usually assume Markovian environments,

Preprint. Under review.

so that they don't need to incorporate memories. However, the environment and instructions are not always Markovian. For example, the instruction "Wave the end effector 3 times" explicitly requires the system to remember how many times it has waved the hand. Secondly, compounding error also grows extremely in this process. Compounding error usually grows significantly above linear as the number of timesteps grows. Such system becomes very unstable in the scenario of long instructions which requires more timesteps.

In this paper, we propose a potential solution for such problems, but still in the realm of imitation learning. The proposed method splits up the agent into 2 separate parts: a translator and an executor. Given a complex command as well as an initial image, the translator takes the responsibility of understanding the command and translate it into detailed, plain and atomic list of commands. In this way, the translator acts as a higher-level commander – it plans what actions to take from a general perspective, and thus the actual action policy does not need to remember any non-markovian information. On the other hand, the executor now only focuses on executing the current sub-task, which downgrades the difficulties to the normal imitation learning level.

In order to evaluate our method, we collect a new imitation learning dataset on TinyUR5 (16), a 2D simulator depicting a tabletop set up, including a UR5 style robot and a few objects that can manipulate with. In this new dataset, we collected 5000 demonstrations of long tasks, each of them including manipulating at least 2 objects and up to 5. Testing on these tasks, we report a non-zero success rate, which is significantly greater than baselines which always have 0% success rate.

In this paper, we summarize the contributions as follows:

- Collecting a long instruction language-conditioned imitation learning dataset.
- Proposing a new method which decreases the difficulty of long instruction imitation learning.
- Tests on long tasks demonstrate non-zero success rate, which significantly outperforms baselines.

2 Methodology

2.1 Problem Statement

Our method tackles the problem under the domain of language-conditioned imitation learning. The goal is to learn a policy $\pi_\theta(a|s, I)$, parameterized by θ, where s refers to a natural language command that includes tasks that include at least 2 separate actions, and I is the current observed image from the agent robot. The policy π generates action a for the current timestep, which is executed in the environment and this process is repeated until the task is accomplished or it fails.

In order to train such system using imitation learning, we firstly create a dataset of expert demonstrations $\mathcal{D} = \{d_0, d_1, ..., d_{n-1}\}$, where $d_i, i \in [0, n)$ is a demonstration composed of a list of consecutive state-action pairs through out all timesteps in this demonstration, i.e., $d = [(s_0, I_0, a_0), (s_1, I_1, a_1), ..., (s_{T-1}, I_{t-1}, a_{T-1})]$. Trained on this dataset, the policy π_θ adapts to the collected dataset and thus acquires the ability to carry out corresponding actions.

2.2 Proposed Method

Overall, our model of the policy contains 2 separate parts – a translator and an executor. The translator takes in the current observation, an image I and a language command s as inputs, and generates a list of detailed sub-commands which only requires only 1 single action to accomplish. For example, given a sentence "Push forward all the food", the translator is supposed to look at the image and find all the food appearing in the image, and correspondingly generate a list of sentences like "Push forward the apple", "Push forward the orange" and so on. Afterwards, this list is treated as inputs of the executor. The executor takes in one sentence and one current image at a time, and generates the actions to execute. After a sentence is finished, the executor follows up to fetch the next sentence as the current sentence. This process is repeated until all sentences from the output of translator are executed.

2.2.1 Translator

The translator is a multimodal deep learning model, which consists of an image encoder, a sentence encoder, a fusion module and a sentence decoder. The general working process starts from the image encoder and sentence encoder, which takes in the current image as well as the general complex language command, and generates an image embedding and a sentence embedding respectively. In our implementation, we choose to use the pretrained image and text encoder from OpenAI's CLIP (8)

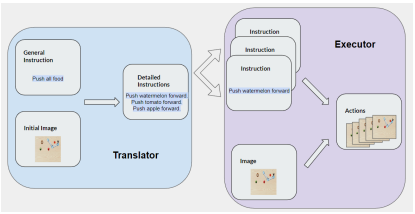

Figure 1: Proposed Method: Translator and Executor

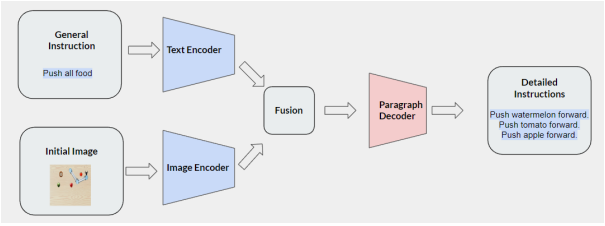

Figure 2: Translator Architecture: The Text Encoder and the Image Encoder are from OpenAI CLIP.

model. Regardless of the dimension of the input images and sentences, the model generates 2 vectors of 1×512 as the image and sentence embeddings respectively.

For fusion purposes, we choose to stay aligned with the pipeline of late fusion (17). After both embeddings are generated, we concatenate them as a larger vector of 1×1024. This operation is also aligned with IBC (18), which fuses action and state into one single vector by concatenation. We argue that although there are plenty of room for testing for a better way of modality fusion, this simple concatenation has demonstrated sufficiently good performances that achieve our goal of reaching a non-zero success rate.

After the model generates a joint embedding space consisting both the language and sentence, we use it to generate a list of detailed sentences describing sub-tasks one by one. Our decoder is supported by a three-layer GRU (19), whose cell state comes from the joint embedding, and generates the sentence in a auto-regressive manner, following many language generation paradigms. In order to mark the start and end of each sentence, we use <SOS> as the start of all tokens, <EOS> as the end of all tokens, and <SEP> as separation tags to divide between each sentences. In order to train the model robustly, we incoporate both teacher forcing (20) and student forcing. While inference, although enabling beam search has the potential for a better translation performance, we also argue the current model has been good enough to reach the goal of this paper, which is achieving a non-zero success rate.

2.3 Executor

The executor is trained on each of the single atomic tasks. For the purpose of fair comparison, we adopt both BC-Z style and ModAttn stype policy network as the executor. Both of them take in the current image and a sentence as input, and spit out the action to execute. Besides, ModAttn also takes in proprioceptions as inputs for a better control of the robot.

2.3.1 BC-Z Executor

BC-Z takes in a simple FiLM style modality fusion of an image and a sentence. In this model, the pipeline is a ResNet, which accepts images as input and generates an embedding. Apart from that, there is another branch of inputs, which is an encoder for the sentence, after generating the embedding of the sentence using a recurrent neural network, the embedding is added up onto specific layers in the ResNet to influence the features. This is a method different from late fusion and has demonstrated good results on many Multi-Modal tasks. Afterwards, the fused embedding is simply sent into a MLP to generate actions.

2.3.2 ModAttn Executor

The ModAttn executor, on the other hand, is an attention and transformer based model. It follows DeTr and Slot Attention style to perform object-centric detection from the image. Different from the original DeTr and Slot Attention, the target object to detect is guided by the encoded language, such that the language provides the model which object to detect, and the model tries to find that in the image. In this method, attention processes are also supervised, which successfully forms modularity, which provides possibility for easy transfer to new robots and environments, and also allows for data-efficient training.

Method	Upper Bound	Naive Long Inst	Translator+Executor
BC-Z	0.53	0.00	0.14
ModAttn	0.89	0.00	0.26

Table 1: Success rates of baselines and our proposed method.

3 Experiments

We build a dataset of long instructions on TinyUR5 simulator. TinyUR5 depicts a table top robot manipulation scenario with a few objects for the robot to play with. The robot has 4 degrees of freedom – 3 free joints, and a gripper. Using this dataset, we defined 5 different long tasks:

- Push both. In this task, the robot is asked to push 2 indicated objects on the table.
- Push food. There will be multiple objects on the table, and the robot need to identify which objects are food, and push all of the food objects forward.
- Push and rotate. The robot is asked to push an object forward and rotate the object.
- Place in a line. Three objects are identified by the sentence. The robot needs to pull these 3 objects into a line.
- Pick place twice. Three objects are also indicated by the sentence. The robot needs to place the first object and the second object onto the third object.

All of the above tasks include as least 2 actions. In order to successfully execute them, the agent needs to understand what actions have been done and what it needs to do currently.

3.1 Translator Performance

Firstly, we test the translator's performance. Given an image and a complex command, the translator spits out a list of sentence, separated by <SEP>. We measure the BLEU score between the generated sentences and ground truth ones collected by the language template to show the quantitative performance. Listed here are a few samples:

- Label: <SOS> push tomato forward <SEP> push strawberry forward <SEP> push apple forward <EOS>; Prediction: <SOS> push apple forward <SEP> push strawberry forward <SEP> push tomato forward <EOS>.
- Label: <SOS> push tomato forward <SEP> push watermelon forward <EOS>; Prediction: <SOS> push milk bottle forward <SEP> push watermelon forward <EOS>.

The BLEU score achieves 90.7 after training. From the samples, we can see that the grammar tag tokens are basically correct. Sometimes it might mistake the order of the objects or detect a wrong object. But overall, substantial convergence and frequent successes are observed.

3.2 Overall Performance

The final success rates are shown in Tab. 1. As can be seen, firstly we train and test BC-Z and ModAttn on a dataset of short atomic actions, depicted in column "Upper Bound". This performance is treated as upperbounds because long tasks will be harder than atomic actions. Afterwards, we train the models directly on the long datasets. Both models are not able to perform the desired behavior correctly and achieve 0% success rate. Finally, we train and test on our proposed translator and executor method. Although still far from the performance of upper bound, we still successfully demonstrate a non-zero success.

4 Conclusion

In this paper, we expand the horizon of language-conditioned imitation learning on to long and non-atomic instructions. Experiments on previous work show failures on these of tasks. By separating the policy agent into a translator and an executor, our proposed method outperforms all of the baselines and for the first time achieves non-zero success.

References

[1] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor, "Language-conditioned imitation learning for robot manipulation tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13139–13150, 2020.

[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*, vol. 33, pp. 1877–1901, 2020.

[3] 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.

[4] 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*, vol. 30, 2017.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[6] 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.

[7] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," *Advances in neural information processing systems*, vol. 32, 2019.

[8] 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*. PMLR, 2021, pp. 8748–8763.

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

[10] Mohit Shridhar, Lucas Manuelli, and Dieter Fox, "Cliport: What and where pathways for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 894–906.

[11] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng, "Code as policies: Language model programs for embodied control," *arXiv preprint arXiv:2209.07753*, 2022.

[12] 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.

[13] 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*. PMLR, 2022, pp. 991–1002.

[14] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018, vol. 32.

[15] Yifan Zhou, Shubham Sonawani, Mariano Phielipp, Simon Stepputtis, and Heni Amor, "Modularity through attention: Efficient training and transfer of language-conditioned policies for robot manipulation," in *6th Annual Conference on Robot Learning*.

[16] "Tinyur5," `https://github.com/yfzhoucs/tiny_ur5`.

[17] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 2, pp. 423–443, 2018.

[18] 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*. PMLR, 2022, pp. 158–168.

[19] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[20] Ronald J Williams and David Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural computation*, vol. 1, no. 2, pp. 270–280, 1989.