Manipulate-Anything: Automating Real-World Robots using Vision-Language Models

Jiafei Duan*
University of Washington

Wentao Yuan*
University of Washington

Wilbert Pumacay
Universidad Católica San Pablo

Yi Ru Wang
University of Washington

Kiana Ehsani
Allen Institute for AI

Dieter Fox
University of Washington
NVIDIA

Ranjay Krishna
University of Washington
Allen Institute for AI

***Abstract*—Large-scale endeavors like RT-1[2] and widespread community efforts such as Open-X-Embodiment [8] have contributed to growing the scale of robot demonstration data. However, there is still an opportunity to improve the quality, quantity, and diversity of robot demonstration data. Although vision-language models have been shown to automatically generate demonstration data, their utility has been limited to environments with privileged state information, they require hand-designed skills, and are limited to interactions with few object instances. We propose Manipulate-Anything, a scalable automated generation method for real-world robotic manipulation. Unlike prior work, our method can operate in real-world environments without any privileged state information, hand-designed skills, and can manipulate any static object. We evaluate our method using two setups. First, Manipulate-Anything successfully generates trajectories for all 5 real-world and 12 simulation tasks, significantly outperforming existing methods like VoxPoser. Second, Manipulate-Anything's demonstrations can train more robust behavior cloning policies than training with human demonstrations, or from data generated by VoxPoser [22] and Code-As-Policies [26]. We believe Manipulate-Anything can be the scalable method for both generating data for robotics and solving novel tasks in a zero-shot setting. Project page: robot-ma.github.io.**

I. Introduction

The success of modern machine learning systems fundamentally relies on the *quantity* [24, 6, 19, 35, 34, 42], *quality* [13, 50, 33, 32, 25], and *diversity* [12, 17, 41, 48, 5] of the data they are trained on. The availability of large-scale internet data made possible significant advances in vision and language [9, 28, 36]. However, the dearth of data has prevented similar advancements in robotics. Human demonstration collection methods do not scale to sufficient *quantity* or *diversity*. Projects like RT-1 [2] demonstrated the utility of high-*quality* human data collected over 17 months. Others have developed low-cost hardware for data collection [7, 43, 10]. However, all these procedures require expensive human collection. In an effort to diversify demonstration data, Open X-Embodiment project collected 1 million trajectories collected through a participatory effort by 34 research labs [8]. Despite the widespread effort, the dataset only contains 20 tasks.

Automated data collection methods do not scale to sufficient *diversity*. With the advent of vision-language models, the robotics community has been abuz with new systems that leverage VLMs to guide robotic behavior [39, 26, 44, 45, 22, 18, 31]. In these systems, VLMs decompose tasks into language plans [39, 26] or generate code to execute predefined skills [20, 22]. Though successful in simulation, these methods underperform in the real world [20, 22]. Some methods rely on privileged state information only available in simulation [44, 18], require hand-designed skills [45]. or are also limited to manipulating a fixed set of object instances with known geometric shape [22, 20].

We propose Manipulate-Anything, a scalable automated demonstration generation method for real-world robotic manipulation. Manipulate-Anything produces high *quality* data, at large-*quantities* (if needed), and can manipulate a *diverse* set of objects to perform a *diverse* set of tasks. When placed in a real world environment and given a task (e.g., "open the top drawer" in Figure 1), Manipulate-Anything effectively leverages VLMs to guide a robotic arm to complete the task. Unlike prior methods, it doesn't need privileged state information, hand-designed skills, or limited to specific object instances. Not relying on privileged information makes Manipulate-Anything environment-agnostic. Thus it can easily generalized to the real world. Manipulate-Anything plans a sequence of sub-goals and generates actions to excute the sub-goals. It can verify whether the robot succeeded in the sub-goal using a verifier and re-plan from the current state if needed. This error recovery enables mistake identification, re-planning, and recovering from failure. It also injects recovery behavior into the collected demonstrations. We further enhanced the VLMs' capabilities by incorporating reasoning from multi-viewpoints, significantly improving performance.

We showcase the utility of Manipulate-Anything through two evaluation setups. First, we show that it can be prompted with a novel, never-before-seen task and complete it in a zero-shot manner. We quantitatively evaluate across 5 real-world and 12 RLBench [23] simulation tasks and demonstrate capabilities across many real-world everyday tasks (refer to supplementary). Our method significantly outperforms VoxPoser [22] in 9/12 simulation tasks for zero-shot evaluation. It also generalizes to tasks where VoxPoser completely fails because of its limitation to specific object instances. Furthermore, we demonstrated

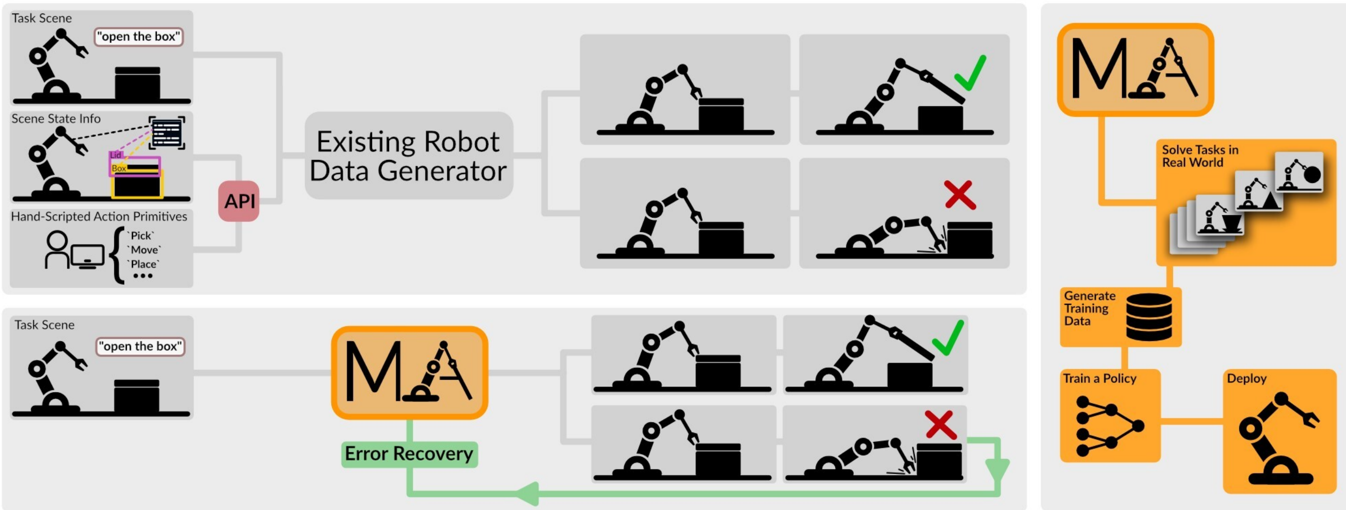

Fig. 1: Manipulate-Anything is an automated method for robot manipulation in real world environments. Unlike prior methods, it doesn't require priviledged state information, hand-designed skills, or limited to manipulating a fixed number of object instances. It can guide a robot to accomplish a diverse set of unseen tasks, manipulating diverse objects. Furthermore, the generated data enables training behavior cloning policies that outperform training with human demonstrations.

that our approach can solve real-world manipulation tasks in a zero-shot manner, achieving a task-averaged success rate of 36%. Second, we show that Manipulate-Anything can generate useful training data for a behavior cloning policy. We compare Manipulate-Anything generated data against ground truth human demonstrations as well as against data from VoxPoser[22] and Code-As-Policies [26]. Surprisingly, policies trained on our data outperforms even human data on 5 out of 12 tasks and performs on par for 3 more. Meanwhile, the baselines are unable to generate the training data for some of tasks. Manipulate-Anything demonstrates the broad possibility of large-scale deployment of robots across unstructured real-world environments. It also highlights its utility as a training data generator, aiding in the crucial goal of scaling up robot demonstration data.

II. Related work

Manipulate-Anything enables scaling of robotic manipulation data using . As such, we review recent efforts in 1) scaling manipulation data, and 2) applications of VLMs in robotics.

Scaling manipulation data. When deploying vision and language-based control policies for real-world applications, a significant challenge revolves around acquiring data. Traditionally, a convenient avenue to collect such trajectories is through human annotations for action (i.e. through teleoperation) and language labeling [38, 37, 3], however, this approach is limited to scale. To address this limitation and achieve autonomous scalability, prior works employ vision-language models or procedurally generate language annotations in simulated environments [18, 11]. For action labels, strategies range from random exploration to learned policies [46]. While human egocentric videos are relevant, they lack action labels and require cross-embodiment transfer [16]. Another strategy involves model-based policies, such as task and motion planning (TAMP) [14]. Our approach extends these methods by incorporating common-sense knowledge from large language models (LLMs) and vision language models (VLMs), by providing a framework which combines the strengths of VLMs, object pose prediction, and dynamic retry to synthesize demonstrations in simulated and real environments.

Language models for robotics. In the field of robotics, large language models have found diverse applications, including policy learning [47], task and motion planning [27, 21], log summarization [30], policy program synthesis [26], and optimization program generation [39]. Previous research has also explored the physical grounding capabilities of these models [22, 20], while ongoing work investigates their integration with task and motion planners to create expert demonstrations [18]. [3] attempted to collect extensive real-world interaction data, with short-horizon trajectories. [29] proposed a key-point based visual prompting method for real-world manipulation, through predicting affordances and corresponding motions. Our work complements the existing line of works, by leveraging the high-level planning capabilities of language models, scene understanding capabilities of vision language models, and action sampling, to enable synthesis of robot trajectories, which include language, vision, and robot state, given arbitrary tasks and environments.

A. Task plan generation

Manipulate-Anything takes as input any task described by a free-form language instruction, $\mathbf{T}$ (e.g., *'open the top drawer'*). Creating robot trajectories that adheres to $\mathbf{T}$ is challenging due to its potential complexity and ambiguity, requiring a nuanced understanding of the current environment state. Given $\mathbf{T}$, and an image of the scene, we apply a VLMs to first identify task-relevant objects in the scene, appending them to a list. Subsequently, We use a VLMs to decompose the main task $\mathbf{T}$ into a series of discrete, smaller sub-goals, represented as $\mathbf{T}_i$, along with the corresponding verification conditions v_i, where i ranges from 1 to n. For the above task, sub-goals include *'grasp the drawer handle'* or *'pull open the drawer'*, and verification conditions are *'did the robot grasp the handle?'* or *'is the drawer opened?'*. This transforms the instruction $\mathbf{T}$ into a sequence of specific sub-goals $\{(\mathbf{T}_1, v_1), (\mathbf{T}_2, v_2), \ldots, (\mathbf{T}_n, v_n)\}$. For each sub-goal, Manipulate-Anything generates desired actions (§ II-B) and uses the corresponding verification condition for each sub-goal to validate whether the generated actions result in the successful completion of the sub-goal (§ II-C). This verification step allows Manipulate-Anything to recover from mistakes and attempt again in the case of failure.

B. Action generation module

Given a sub-goal, the desired output from the action generation module is a sequence of low-level actions represented as a 6 DoF end-effector pose. The actions can be categorized into two sets: agent-centric or object-centric. Agent-centric actions modify the agent's state; e.g., it can move the robot's end-effector from the current state (e.g., "rotate 90°"). We feed the VLMs with the current observation along with in-context learning technique to write a code to synthesize the desired motion. Unlike prior methods that use only language models to generate code [26], our approach utilizes VLMs to understand and reason about object locations and the scene, which helps to ground the generation in the current state of the scene. This advantage is demonstrated in the ablation studies in the Appendix.

Object-centric actions require manipulating a certain object (e.g., "grasp a knife"). We use an **object-agnostic grasp prediction model** [49]. The grasp model generates all the possible 6-DOF grasping poses in the scene. These poses are not conditioned on the objects and could contain errors. From the RGB-D image of the current state, we extract a raw 3D point cloud. The point cloud is sent to the grasp model, which predicts 6-DoF grasps placements across the scene. We then further filter the proposed candidate grasp pose using VLMs with in-context learning and condition it on the given task (e.g., if the task is "grasp a knife", the VLMs will detect the handle of the knife). Lastly, we use the detected bounding box to filter and sample an ideal grasp pose.

A single view point might be insufficient to provide the VLMs with enough information to perform the task (e.g., some views might be occluded by the robot arm). Therefore, for both agent-centric and objec-centric action generation, we **render multiple viewpoints of the scene and query VLMs to choose an ideal viewpoint** given the sub-task. For example, if the task is to open a drawer, the view in which the handle of the drawer is visible would be preferred. After the best view point is chosen, the grasping poses can be filtered limited to the poses visible in that view point or the code generation will be conditioned on that image. After the action is generated, a simple motion planner can be used to move the robot to the desired pose as shown in detailed in Fig. 2.

C. Sub-goal verification

To ensure that each sub-goal T_i is executed correctly, we introduce a -based verifier. After every action for each sub-goal are executed, we use the VLMs to check if the end state matches the verifier condition v_i. Similar to the action generation module, we use **multi-view VLMs reasoning** to find the optimal view, avoiding errors due to occlusion or ambiguity from a single viewpoint. If the verifier identifies failure, we re-attempt the action generation step from the current state. Otherwise, the next sub-goal T_{i+1} is attempted. More details of the implementation is in the Appendix.

III. EXPERIMENTS

Our experiments are designed to address two questions: 1) Can Manipulate-Anything accurately solve a diverse set of tasks in a zero-shot manner? 2) Can data generated from Manipulate-Anything be used to train a robust policy?

Implementation details. We use both GPT-4V and Qwen-VL [1] as our . We use GPT-4V for task decomposition, action generation, and verification. We use Qwen-VL to detect and extract object information. To ensure zero-shot execution within a reasonable budget, we limit the number of action steps in each trajectory to 50 and the verification module allows a maximum of 30 tries to accomplish a sub-goal. For the task plan generation, we follow the prompting structure adapted from ProgPrompt [39]. All prompts input into the VLMs are accompanied by few-shot demonstrations [4]. Additionally, we provide three manually curated primitive action code snippets as examples to prompt the VLMs for new action code generation. Full prompts are included in the Appendix. We use four viewpoints $\mathbf{M}_4 = [front, wrist, left\_shoulder, right\_shoulder]$ for the simulation experiments, and re-render three viewpoints for the real-world experiments [15]. For better reasoning by the , we use a resolution of 256×256.

A. Zero-shot Performance in Simulation

We empirically study the zero-shot capability of Manipulate-Anything in solving 12 diverse tasks in simulation. Our simulation experiments are reported to ensure reproducibility and provide a benchmark for future methods.

Environment and tasks. The simulation is set up in CoppeliaSim and interfaced through PyRep. All simulation experiments use a Franka Panda robot with a parallel gripper. Input observations are captured from four RGB-D cameras positioned around a tabletop setting. We use RLBench [23], a robot learning benchmark with diverse tasks conditioned on language and provided success conditions. We sample 12 tasks from RLBench, covering a diverse range of action primitives, task horizons, and object position perturbations. Each action can be represented as a way-point, and the trajectories are computed and executed via a motion planner using the Open Motion Planning Library[40].

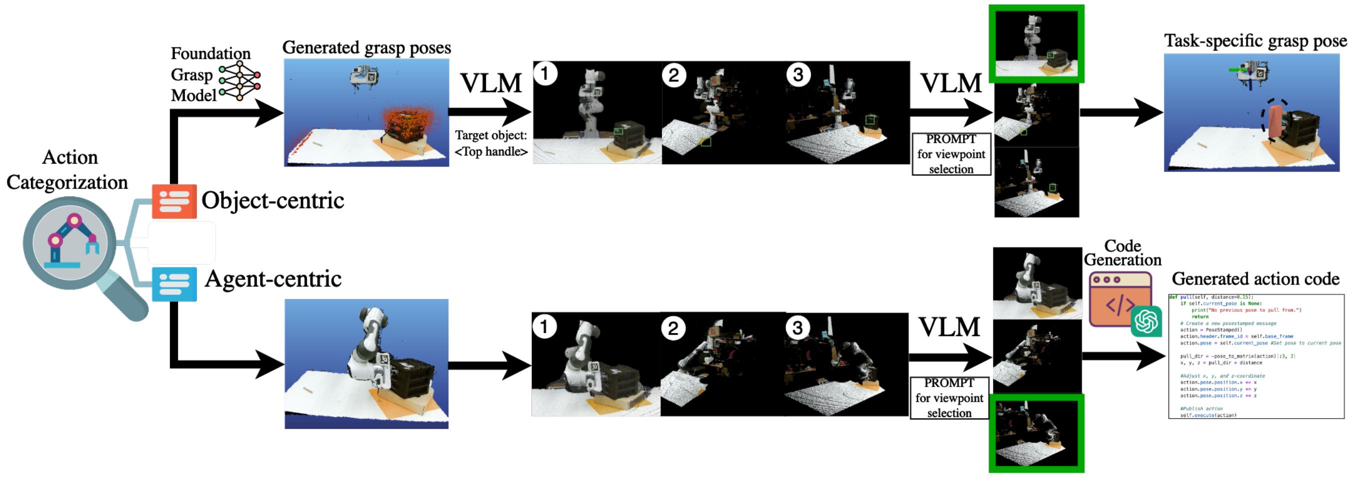

Fig. 2: **Action Generation Module.** Manipulate-Anything enables generation of two types of actions: object-centric and agent centric. For object-centric actions which require manipulation of an object, we leverage a foundation grasp model to generate all suitable grasps. Next, we leverage a VLM to detect the object from mult-view frames, and along with the candidate grasp poses and target subgoal, query the VLMs to select the best view point. We filter and select the optimal grasp for the sub-goal. For more agent-centric actions, the view-point selection process is the same, and the goal is to output code representing the change in pose of the end-effector from the current frame.

TABLE I: **Task-averaged success rate % for zero-shot evaluation.** outperformed other baselines in 9 out of 12 simulation tasks from RLBench [23]. Each task was evaluated over 3 seeds to obtain the task-averaged success rate and standard deviations.

Method	Put_block	Play_jenga	Open_jar	Close_box	Open_box	Pickup_cup
VoxPoser [22]	70.7±2.31	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00	26.7±14.00
CAP [26]	84.00±16.00	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00	14.67±4.62
MA (Ours)	**96.00**±4.00	**77.33**±6.11	**80.00**±4.00	**33.33**±12.86	**29.00**±10.07	**82.67**±14.04
Method	Take_umbrella	Sort_mustard	Open_wine	Lamp_on	Put_knife	Pick_&_lift
VoxPoser[22]	33.33±8.33	**96.0**±6.93	8.00±4.00	57.3±12.22	**92.00**±4.00	96.00±0.00
CAP[26]	4.00±4.00	0.00±0.00	0.00±0.00	64.00±6.93	14.67±8.33	**100.00**±0.00
MA (Ours)	**61.33**±20.13	64.00±6.93	**42.00**±4.00	**69.33**±6.11	52.00±10.58	84.00±6.93

Baselines. We compare against two state-of-the-art zero-shot data generation approaches: Code-as-Policies (CAP) [26] and VoxPoser [22]. CAP uses language models to generate executable programs that call hand-crafted primitive actions. VoxPoser [22] builds a 3D voxel map of value functions for predicting waypoints. We provide both CAP and VoxPoser with ground truth simulation state information of the target object's asset names or positions.

Results: Manipulate-Anything can generate successful trajectories for all 12 **tasks while VoxPoser and CAP cover only** 8 **and** 6 **tasks, respectively** (Table I). Without the privileged state information, the baselines would not succeed on any of the 12 tasks. Manipulate-Anything outperforms the baselines in 9 out of the 12 tasks. The three tasks where our method achieves lower performance require fine-grained manipulation of objects, which are the hardest task without the privileged state information used by baselines. VoxPoser fails in the tasks that require moving the arm beyond 4-DoF. Manipulate-Anything outperforms the strongest baseline, VoxPoser, by an average task-averaged margin of up to 25%.

B. Behavior cloning with demonstrations from Manipulate-Anything

Next, we analyze the quality of the generated data by comparing the success rates of behavior cloning models trained with the data. Zero-shot methods like Manipulate-Anything are computationally expensive but hold the potential to generate useful training data. To evaluate the quality and effectiveness of the generated training data, we use the methods described in the previous section to generate data for each task. We also compare performance against a model trained on human-generated demonstrations across the 12 tasks. We use the data to train behavior cloning policies.

Data generation details. We generate 10 successful demonstrations per task. We use the system's success condition to filter for successful demonstrations. Each of the demonstrations consist of a language instruction, RGB-D frames for the trajectory, and waypoints represented as 6 DoF gripper poses and states. For the tasks that the baselines were unable to generate any successful demonstrations, we patched the missing training data with RLBench system-generated demonstrations.

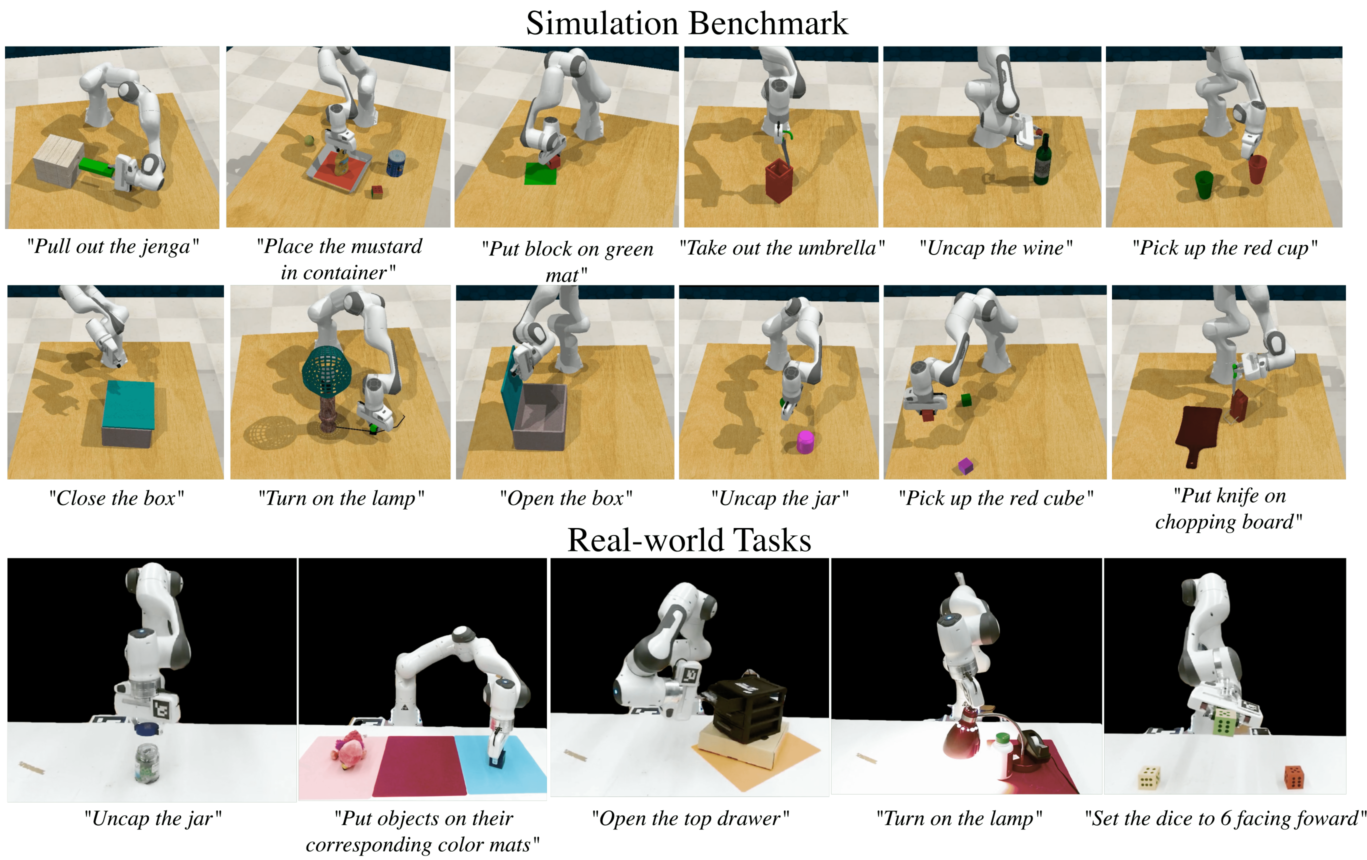

Fig. 3: Manipulate-Anything is an open-vocabulary autonomous robot demonstration generation system. We show zero-shot demonstrations for 12 tasks in simulation 5 tasks in the real world.

Training and evaluation protocol. Using the generated demonstrations, we train a Perceiver-Actor (PerAct) model, which is a transformer-based robotic manipulation behavior cloning model [38]. The model expects tokenized voxel grids and language instructions as inputs and predicts discretized voxel grid 6 DoF poses and gripper states. For all the generated training datasets, we train a multi-task PerAct policy with a batch size of 4 for 30k iterations. To ensure consistent evaluation, we generate one set of testing environments with RLBench. We evaluate the last checkpoint from each of the trained policies. Each policy is evaluated for 25 episodes across each task using 3 different seeds. We measure the success rate based on the simulation-defined success condition.

Results: Policies trained using Manipulate-Anything data perform similarly to policies trained using human demonstrations ($p = 0.973$) **(Table II)**. Training on either Manipulate-Anything or on human demonstrations results in a performance difference of a mere 0.27% across all tasks. Furthermore, models trained on data from the baselines exhibit a statistically lower performance ($p \leq 0.01$ for both VoxPoser and CAP). One of the main factors potentially contributing to the differences in the performance could be that Manipulate-Anything generates diverse expert trajectories that are preferable to humans. This can be seen in Fig. 4, which shows the action distribution of the generated data by different methods for the same given tasks. Additionally, our generated data recorded the lowest Chamfer Distance (CD) of 0.056 with human-generated demonstrations data. We also observed that the policy trained on MA data achieves a lower standard deviation of 3.39 across all tasks compared tozero-shot performance of 8.48. This suggests the benefits of training over generated data instead of relying solely on zero-shot deployment.

C. Real-world experiments

Finally, we evaluate Manipulate-Anything in the real world. We also automatically generate real-world demonstrations for training PerAct.

Environment and tasks. We employ a Franka Panda manipulator equipped with a parallel gripper. We use a front-facing Kinect 2 RGB-D camera. To generate multi-view inputs for the Manipulate-Anything framework, we re-render virtual viewpoints from the generated point cloud, similar to prior work [15]. We select 5 representative real world tasks: `open_jar`, `sort_objects`, `correct_dices`, `open_drawer`, and `on_lamp`, all conditioned on language instructions. We evaluate each task for 10 episodes, with varying object poses across 3 trials of evaluation.

Data generation details. We used Manipulate-Anything to generate 6 demonstrations for each task and manually perform

TABLE II: **Behavior Cloning with different generated data.** The behavior cloning policy trained on the data generated by provides the best performance on 10 out of 12 tasks compared to the other autonomous data generation baselines. We report the Success Rate % for behaviour cloning policies trained with data generated from VoxPoser [22] and Code as Policies [26] in comparison. Note that the RLBench[23] baseline uses human expert demonstrations and is considered an upper bound for behavior cloning.

Generated data	`Put_block`	`Play_jenga`	`Open_jar`	`Close_box`	`Open_box`	`Pickup_cup`
VoxPoser[22]	2.67±2.31	-	-	-	-	4.00±4.00
CAP[26]	6.67±2.31	-	-	-	-	14.67±12.86
MA (Ours)	**85.33**±10.07	**81.33**±2.31	21.33±10.07	42.67±8.33	**30.67**±11.55	54.00±12.49
RLBench[23]	20.00±18.33	**81.33**±9.24	**58.67**±45.49	**68.00**±24.98	14.67±6.11	**54.67**±23.09

Generated data	`Take_umbrella`	`Sort_mustard`	`Open_wine`	`Lamp_on`	`Put_knife`	`Pick_&_lift`
VoxPoser[22]	4.00±4.00	0.00±0.00	1.33±2.31	5.33±4.62	1.33±2.31	5.67±1.64
CAP[26]	13.33±10.06	-	-	8.00±16.00	9.33±6.11	46.67±2.31
MA (Ours)	**84.00**±6.93	**53.33**±6.11	**86.67**±6.11	**89.33**±6.11	8.00±4.00	33.33±2.31
RLBench[23]	58.67±50.80	**53.33**±34.02	**86.67**±12.86	84.00±13.86	**30.67**±10.07	**62.67**±9.24

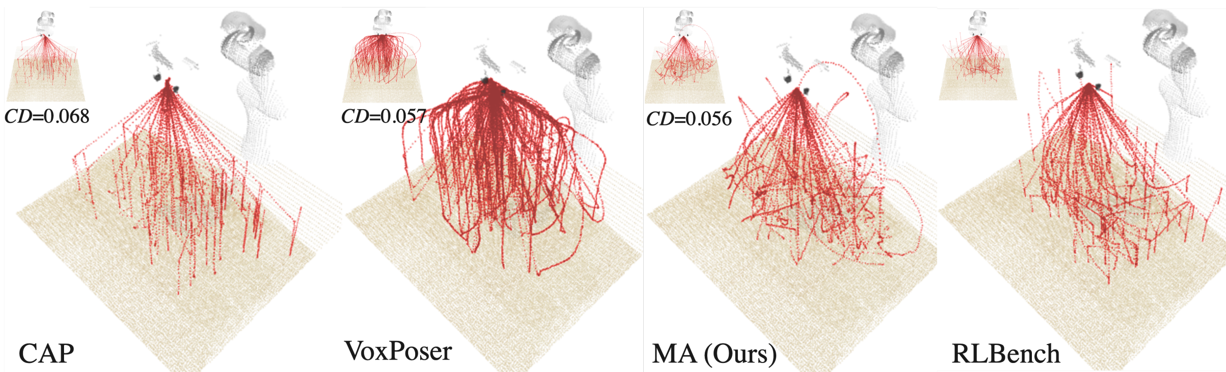

Fig. 4: **Action Distribution for Generated Data:** We compare the action distribution of data generated by various methods against human-generated demonstrations via RLBench on the same set of tasks. We observed a high similarity between the distribution of our generated data and thehuman-generated data. This is further supported by the computed CD between our methods and the RLBench data, which yields the lowest (CD=0.056).

scene resets when failures occur. We train a similar multi-task PerAct for 120k iterations and evaluate the trained policies in a manner similar to the zero-shot experiments.

Results: Manipulate-Anything is able to generate successful demonstrations for each of the 5 **real world tasks.** Even for the worst-performing task, Manipulate-Anything achieves a success rate of more than 25%. Our approach outperforms **CAP** by 38%. Consistent with the simulation results, **training with the data generated by Manipulate-Anything produces a more robust policy** compared to performing zero-shot. Additionally, in 4 out of 5 tasks, the trained policies perform better than the zero-shot approach. The policy underperforms on the `sort_object` task, because it requires longer-horizon memory —a known limitation pointed out in PerAct [38].

D. Ablations

For effective real-world deployment of Manipulate-Anything, it's crucial that the collected data supports scaling of robotics transformers and offers diverse skills and interacted objects. We conducted an ablation study to evaluate the quality of Manipulate-Anything-generated data for scaling and its generalization to language instruction changes. For scaling, we generated behavior cloning data, ranging from 1 to 100 training demonstrations from RLBench and Manipulate-Anything for a single task, and trained a PerAct policy. For generalization, we varied the `sort_mustard` task with different language instructions and target objects. We compared our approach to VoxPoser to assess robustness to object and language instruction changes. Further implementation details are in the supplementary materials. **Result:** Our scaling experiments demonstrate that generating more training data via Manipulate-Anything improves PerAct policy performance (Fig. 5). The data from our approach shows a better rate of change with a slope of 0.503 for a linear fit, compared to 0.197 for RLBench-generated data. Additionally, Manipulate-Anything data is more generalizable and robust to language instruction changes, outperforming VoxPoser in task success across language and object variations. Detailed results in the appendix.

TABLE III: **Real-world Results.** The model trained on the data generated by our model in the real world (no expert in the loop) demonstrates on par results with the model trained on human expert collected data. We present a comparison of success rates for task completion in a zero-shot manner (Code as Policies [26] and), and using trained policies from data and human expert data.

	Open_drawer	Sort_object	On_lamp	Open_jar	Correct_dice
CAP (0-shot)	0.00 ± 0.00	13.33 ± 5.77	0.00 ± 0.00	6.67 ± 5.77	6.67 ± 5.77
MA (0-shot)	**36.67**±5.77	**60.00**±10.00	**26.67**±11.55	**40.00**±10.00	**53.33**±5.77
PerAct (MA data)	50.00 ± 0.00	33.33 ± 5.77	46.67 ± 5.77	56.67 ± 5.77	60.00 ± 0.00
PerAct (Human data)	**53.33**±11.55	**36.67**±5.77	**60.00**±0.00	**76.67**±5.77	**80.00**±10.00

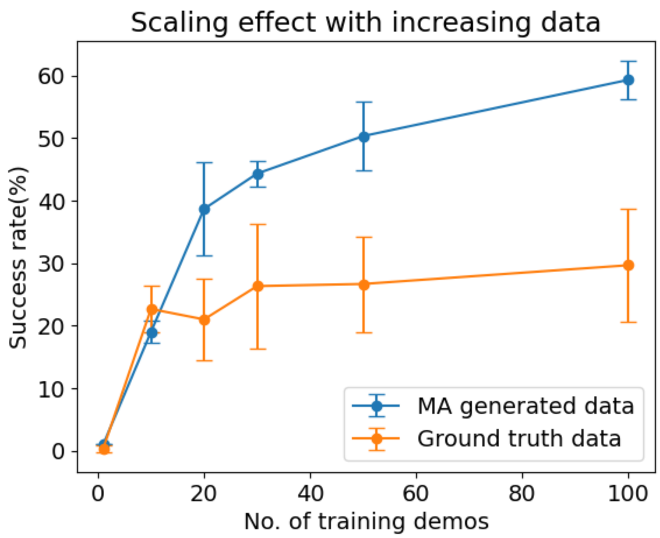

Fig. 5: **Scaling experiment.** Scaling effect of model performance with increasing training demonstrations.

IV. Discussion

Limitations. Manipulate-Anything relies on the availability of . While this can pose a dependence on the foundational models with the rise of the open , we believe this issue will be addressed soon. **Future work.** With the enhancements of large foundational models, Manipulate-Anything, due to its modularity, will continue to grow and scale up to more complex tasks. **Conclusion.** Manipulate-Anything is a scalable environment-agnostic approach for generating 0-shot demonstration for robotic tasks without the use of privileged environment information. Manipulate-Anything uses VLMs to do high level planning and scene understanding and is capable of error recovery. This enables high quality data generation for behavior cloning that can achieve better performance that using human data.

References

[1] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023.

[2] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[3] Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on Robot Learning*, pages 287–318. PMLR, 2023.

[4] 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.

[5] Zeming Chen, Alejandro Hernández Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, et al. Meditron-70b: Scaling medical pretraining for large language models. *arXiv preprint arXiv:2311.16079*, 2023.

[6] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829, 2023.

[7] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

[8] Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models. https://arxiv.org/abs/2310.08864, 2023.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Jiafei Duan, Yi Ru Wang, Mohit Shridhar, Dieter Fox, and Ranjay Krishna. Ar2-d2: Training a robot without a

robot. *arXiv preprint arXiv:2306.13818*, 2023.

[11] Kiana Ehsani, Tanmay Gupta, Rose Hendrix, Jordi Salvador, Luca Weihs, Kuo-Hao Zeng, Kunal Pratap Singh, Yejin Kim, Winson Han, Alvaro Herrasti, et al. Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[12] Alex Fang, Gabriel Ilharco, Mitchell Wortsman, Yuhao Wan, Vaishaal Shankar, Achal Dave, and Ludwig Schmidt. Data determines distributional robustness in contrastive language image pre-training (clip). In *International Conference on Machine Learning*, pages 6216–6234. PMLR, 2022.

[13] Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. Datacomp: In search of the next generation of multimodal datasets. *Advances in Neural Information Processing Systems*, 36, 2024.

[14] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293, 2021.

[15] Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conference on Robot Learning*, pages 694–710. PMLR, 2023.

[16] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[17] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020.

[18] Huy Ha, Pete Florence, and Shuran Song. Scaling up and distilling down: Language-guided robot skill acquisition. *arXiv preprint arXiv:2307.14535*, 2023.

[19] 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.

[20] Haoxu Huang, Fanqi Lin, Yingdong Hu, Shengjie Wang, and Yang Gao. Copa: General robotic manipulation through spatial constraints of parts with foundation models. *arXiv preprint arXiv:2403.08248*, 2024.

[21] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[22] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[23] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[24] 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.

[25] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*, 2021.

[26] 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. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[27] Kevin Lin, Christopher Agia, Toki Migimatsu, Marco Pavone, and Jeannette Bohg. Text2motion: From natural language instructions to feasible plans, 2023.

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[29] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[30] Zeyi Liu, Arpit Bahety, and Shuran Song. Reflect: Summarizing robot experiences for failure explanation and correction, 2023.

[31] Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, Quan Vuong, Tingnan Zhang, Tsang-Wei Edward Lee, Kuang-Huei Lee, Peng Xu, Sean Kirmani, Yuke Zhu, Andy Zeng, Karol Hausman, Nicolas Heess, Chelsea Finn, Sergey Levine, and Brian Ichter. Pivot: Iterative visual prompting elicits actionable knowledge for vlms, 2024.

[32] Thao Nguyen, Gabriel Ilharco, Mitchell Wortsman, Sewoong Oh, and Ludwig Schmidt. Quality not quantity: On the interaction between dataset design and robustness of clip. *Advances in Neural Information Processing Systems*, 35:21455–21469, 2022.

[33] Thao Nguyen, Samir Yitzhak Gadre, Gabriel Ilharco, Se-

woong Oh, and Ludwig Schmidt. Improving multimodal datasets with image captioning. *Advances in Neural Information Processing Systems*, 36, 2024.

[34] Vivek Ramanujan, Thao Nguyen, Sewoong Oh, Ali Farhadi, and Ludwig Schmidt. On the connection between pre-training data diversity and fine-tuning robustness. *Advances in Neural Information Processing Systems*, 36, 2024.

[35] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

[36] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=M3Y74vmsMcY.

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

[38] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[39] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023.

[40] Ioan A Sucan, Mark Moll, and Lydia E Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012.

[41] Yonglong Tian, Lijie Fan, Kaifeng Chen, Dina Katabi, Dilip Krishnan, and Phillip Isola. Learning vision from models rivals learning vision from data. *arXiv preprint arXiv:2312.17742*, 2023.

[42] Vishaal Udandarao, Ameya Prabhu, Adhiraj Ghosh, Yash Sharma, Philip HS Torr, Adel Bibi, Samuel Albanie, and Matthias Bethge. No" zero-shot" without exponential data: Pretraining concept frequency determines multimodal model performance. *arXiv preprint arXiv:2404.04125*, 2024.

[43] Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C Karen Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.

[44] Lirui Wang, Yiyang Ling, Zhecheng Yuan, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, and Xiaolong Wang. Gensim: Generating robotic simulation tasks via large language models. *arXiv preprint arXiv:2310.01361*, 2023.

[45] Yufei Wang, Zhou Xian, Feng Chen, Tsun-Hsuan Wang, Zackory Erickson, David Held, and Chuang Gan. Genbot: Generative simulation empowers automated robotic skill learning at scale. *arXiv*, 2023.

[46] Yufei Wang, Zhou Xian, Feng Chen, Tsun-Hsuan Wang, Yian Wang, Katerina Fragkiadaki, Zackory Erickson, David Held, and Chuang Gan. Robogen: Towards unleashing infinite data for automated robot learning via generative simulation. *arXiv preprint arXiv:2311.01455*, 2023.

[47] Tianbao Xie, Siheng Zhao, Chen Henry Wu, Yitao Liu, Qian Luo, Victor Zhong, Yanchao Yang, and Tao Yu. Text2reward: Automated dense reward function generation for reinforcement learning. *arXiv preprint arXiv:2309.11489*, 2023.

[48] Hu Xu, Saining Xie, Xiaoqing Ellen Tan, Po-Yao Huang, Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. Demystifying clip data. *arXiv preprint arXiv:2309.16671*, 2023.

[49] Wentao Yuan, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. M2t2: Multi-task masked transformer for object-centric pick and place, 2023.

[50] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36, 2024.