Journal of Artificial Intelligence Research 75 (2022) 1595-1631 Submitted 03/2022; published 12/2022

TOOLTANGO: Common Sense Generalization in Predicting Sequential Tool Interactions for Robot Plan Synthesis

Shreshth Tuli S.TULI20@IMPERIAL.AC.UK
Department of Computing
Imperial College London, UK

Rajas Bansal RAJASB@STANFORD.EDU
Department of Computer Science
Stanford University, USA

Rohan Paul ROHAN@CSE.IITD.AC.IN
Mausam MAUSAM@CSE.IITD.AC.IN
Department of Computer Science and Engineering
Yardi School of Artificial Intelligence
Indian Institute of Technology Delhi, India

Abstract

Robots assisting us in environments such as factories or homes must learn to make use of objects as tools to perform tasks, for instance, using a tray to carry objects. We consider the problem of learning common sense knowledge of when a tool may be useful and how its use may be composed with other tools to accomplish a high-level task instructed by a human. Specifically, we introduce a novel neural model, termed TOOLTANGO, that first predicts the next tool to be used, and then uses this information to predict the next action. We show that this joint model can inform learning of a fine-grained policy enabling the robot to use a particular tool in sequence and adds a significant value in making the model more accurate. TOOLTANGO encodes the world state, comprising objects and symbolic relationships between them, using a graph neural network and is trained using demonstrations from human teachers instructing a virtual robot in a physics simulator. The model learns to attend over the scene using knowledge of the goal and the action history, finally decoding the symbolic action to execute. Crucially, we address generalization to unseen environments where some known tools are missing, but unseen alternative tools are present. We show that by augmenting the representation of the environment with pre-trained embeddings derived from a knowledge-base, the model can generalize effectively to novel environments. Experimental results show at least 48.8-58.1% absolute improvement over the baselines in predicting successful symbolic plans for a simulated mobile manipulator in novel environments with unseen objects. This work takes a step in the direction of enabling robots to rapidly synthesize robust plans for complex tasks, particularly in novel settings.

1. Introduction

Advances in autonomy have enabled robots to enter human-centric domains such as homes and factories where we envision them performing general purpose tasks such as transport, assembly, and clearing. Such tasks require a robot to interact with objects, often using them as *tools*. For example, a robot asked to "take fruits to the kitchen", can use a *tray* for carrying items, a *stick* to fetch objects beyond physical reach and may use a *ramp* to reach elevated platforms. Previous work has shown that the ability to predict the possible use of tools for a given task is often useful in guiding a robot's task planner towards plans likely to be feasible (Driess, Oguz, Ha, & Toussaint,

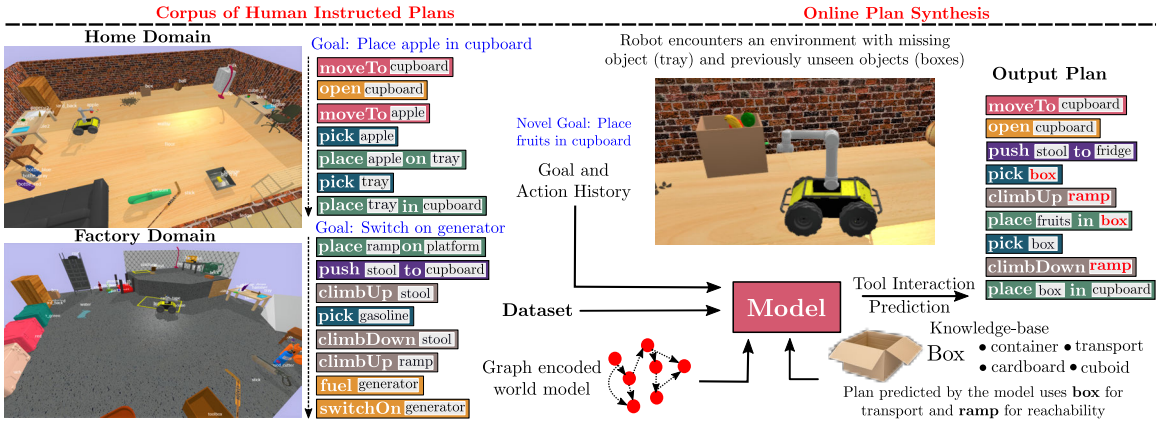


Figure 1: TOOLTANGO acquires commonsense knowledge from human demonstrations leveraging graph-structured world representation, knowledge-based corpora and goal-conditioned attention to perform semantic tasks. Our aim is to acquire commonsense knowledge to develop a generalized goal-conditioned policy for a robot.

2020; Garrett et al., 2021; Choi, Langley, & To, 2018; Levihn & Stilman, 2014; Fitzgerald, Goel, & Thomaz, 2021). In this work we consider the problem of predicting *which* objects could be used as tools and *how* their use can be composed for a task. In essence, we focus on the ability to predict appropriate tools that can guide the robot towards feasible and efficient plans and delegate the issue of dexterous tool manipulation to prior work (Choi et al., 2018; Myers, Teo, Fermüller, & Aloimonos, 2015).

Learning to predict task-directed tool interactions poses several challenges. First, real environments (a household or factory-like domain) are typically large where an expansive number of tool interactions may be possible (e.g., objects supporting others while transporting). Acquiring data for all feasible tool objects or exploring the space of tool interactions is challenging for any learner. Second, the usefulness of a tool varies with context. For example, placing milk in the cupboard may require the robot to elevate itself vertically using a ramp if the milk is placed at a height unreachable by the robot, but if the milk is kept on a table, a simple tray might suffice. Third, the robot may encounter new environments populated with novel objects not encountered during training. Hence, the agent's model must be able to *generalize* by reasoning about interactions with novel objects unseen during training. Humans possess innate commonsense knowledge about contextual use of tools for an intended goal (Allen, Smith, & Tenenbaum, 2019). For example, a human actor when asked to move objects is likely to use trays, boxes, or even improvise with a new object with a flat surface. We aim at providing this commonsense to a robotic agent, so that it can generalize its knowledge to unseen tools, based on shared context and attributes of seen tools (see Figure 1).

This paper takes a step in the direction of enabling robots to learn how to perform high-level tasks such as compositional tool use in semantic tasks, particularly in novel environments. This paper makes four main contributions. As the first contribution, we present a crowd-sourced dataset of human-instructed plans where a human teacher guides a simulated mobile manipulator to leverage objects as tools to perform multi-step actions such as assembly, transport and fetch tasks. The process results in a corpus of $\sim 1,500$ human demonstrated robot plans involving diverse goal settings and environment scenes.

Second, the dataset mentioned above is first used to supervise a (1-step) neural imitation learner that predicts tool applicability given the knowledge of the world state and the intended goal. We show how learning a dense embedding for the environment and that of a background knowledge base can enable the model to generalize to novel scenes with new object instances that may share semantic attributes with objects seen during training; a common problem in state of the art task and policy learning approaches. We introduce a graph neural architecture, TOOLNET, that encodes both the metric and relational attributes of the world state as well as available taxonomic resources such as ConceptNet (Speer, Chin, & Havasi, 2017). The TOOLNET model predicts tool use by learning an attention over entities that can potentially serve as tools. Implicitly, the model acquires knowledge about primitive spatial characteristics (typically an output of a mapping system) and semantic attributes (typically contained in taxonomic resources) enabling generalization to novel contexts with previously unseen objects.

Third, we present an imitation learner that uses the same dataset to make action predictions towards the intended goal. We term the model, Tool Interaction Prediction Network for Generalized Object environments (TANGO). Similar to TOOLNET, TANGO also encodes the world state using a graph neural network and learns to attend over the scene using knowledge of the goal and the action history, finally decoding the symbolic action to execute. The action predictions are interleaved with physics simulation (or execution) steps, which ameliorates the need for modeling the complex effects of actions inherent in tool interactions.

As a final contribution, we combine the two models TOOLNET and TANGO in a single architecture. Specifically, the joint model first makes the predictions for the next tool, and then uses this information to make a better action prediction. We show that this can inform learning of a fine-grained policy enabling the robot to use a particular tool in sequence and adds a significant value in making the model more accurate. We term this joint model TOOLTANGO.

Experimental evaluation with a simulated mobile manipulator demonstrates (a) accurate prediction of tool interaction sequences with high executability/goal-attainment likelihood, (b) common sense generalization to novel scenes with unseen object instances, and (c) robustness to unexpected errors during execution. Additionally, compared to TANGO, we demonstrate the benefit of using TOOLNET predictions for specific complex settings requiring multiple tools to reach the goal state. Experiments show that in previously seen settings, TOOLTANGO gives an absolute improvement of 3.38-5.59% in reaching a goal state for a simulated mobile manipulator compared to the state-of-the-art TANGO model. In unseen settings, the unified model gives 2.48-3.58% improvement compared to TANGO. In comparison to a simple affordance prediction approach, the proposed model performs better in learning the pre-conditions for tool use. Further, the use of a neural model enables higher goal-reaching performance and faster training compared to a vanilla reinforcement learning approach.

A preliminary version of TOOLNET that performs single tool prediction using only the initial environment state was presented as part of the Workshop on Advances & Challenges in Imitation Learning in Robotics at Robotics Science and Systems (RSS) 2020 conference (Bansal, Tuli, Paul, & Mausam, 2020). Our TANGO model was presented as part of the International Joint Conference in Artificial Intelligence (IJCAI) 2021 (Tuli, Bansal, Paul, & Mausam, 2021). This journal submission presents a substantially detailed exposition of the TOOLNET and TANGO models and includes additional supporting background material. Moreover, it also extends TOOLNET to predict *next* tool at each step of a multi-step action execution, instead of a single tool for the whole sequence, as in

the original paper. This paper also combines the two models into TOOLTANGO, and establishes its superior performance.

The remainder of the paper is organized as follows. Section 2 overviews related work. Section 3 formulates the problem of predicting tool interaction sequences. Section 4 details the technical approach and presents the TOOLTANGO model. The data collection platform and the dataset in detailed in Section 5. Section 6 provides the experimental details and results. Finally, Sections 7 and 8 summarizes the work and lays out avenues for future work. The associated code, data set and videos are available at `https://github.com/reail-iitd/tango`. Links to all supplementary material are given in Appendix A.

2. Related Work

2.1 Classical Planning

Reaching goal states from a given world state through symbolic actions is closely tied to the domain of *task and motion planning* (TAMP) (Garrett et al., 2021). Literature presents a large volume of work in this domain, ranging from constraint satisfaction to search based methods. For instance, Toussaint, Allen, Smith, and Tenenbaum (2018) a planner to compose physics tool interactions using a logic-based symbolic planner. Similarly, Srivastava, Fang, Riano, Chitnis, Russell, and Abbeel (2014) provide an implementation agnostic interface between a task and motion planner to combine both for goal-directed planning. Garrett, Lozano-Pérez, and Kaelbling (2020) extend PDDL descriptions to include generic and declarative specifications enabling the planning to be domain independent. These methods only utilize the task properties through action constraints. For instance, the knowledge that a tray can carry multiple objects is only encoded through the constraint that an object can be placed on it. This allows such methods to create *any* feasible plan and not leverage commonsense knowledge to ensure that a goal state is reached in a few steps. Some recent works such as by Silver et al. (2021) aim to *learn* object importance through human demonstrations. However, such methods can only work on objects seen previously in training and cannot generalize to unseen objects. We consider a part of the broad TAMP framework that focuses on determining the set of actions that are likely to take an agent to the goal state while taking the motion planning as a behavioural routine. We build our work on the observation that in domain with a large number of states and possible interactions, the task planning itself becomes challenging. We consider home and factory-like domains inspired from VirtualHome (Puig, Ra, Boben, Li, Wang, Fidler, & Torralba, 2018) and similar related works. We ensure that in our domains the number of objects is large and they can be contained within/supported or transported by other objects as tools (details in Section 5). Similar to the work by Silver et al. (2021), our model learns to prune-away irrelevant objects, additionally considering a domain with richer inter-object interactions. Consequently, our learner makes additional use of semantic properties and exploits correlations between actions and outputs interactions that are likely to lead to successful plans.

2.2 Learning Tool Manipulation Skills

Learning control policies for manipulating tools has received recent attention in robotics. Finn, Yu, Zhang, Abbeel, and Levine (2017) and Park, Noseworthy, Paul, Roy, and Roy (2019) learn tool manipulation policies from human demonstrations. Xie, Ebert, Levine, and Finn (2019) and Wu, Yildirim, Lim, Freeman, and Tenenbaum (2015) learn physics models and *effects* enabling

goal-directed compositional use. Liu, Freeman, Tenenbaum, and Wu (2018) address the problem of learning primitive physical decomposition of tool like object through its physical and geometric attributes enabling their human-like use. Wu, Lim, Zhang, Tenenbaum, and Freeman (2016) learn physical properties of objects from unlabeled videos. Nair, Chen, Agrawal, Isola, Abbeel, Malik, and Levine (2017) and Lynch, Khansari, Xiao, Kumar, Tompson, Levine, and Sermanet (2020) learn to interact with objects in a self-supervised setup. Efforts such as Holladay, Lozano-Pérez, and Rodriguez (2019), and Antunes, Saponaro, Dehban, Jamone, Ventura, Bernardino, and Santos-Victor (2015) plan tool interactions modeling contact and force interactions. Another set of efforts focuses on incorporating the ability to discover the use of objects as tools and using them for plan completion. Rich symbolic architectures such as ICARUS (Choi et al., 2018), KDAP (Kroemer, Ugur, Oztop, & Peters, 2012) and ROAR (Lee, Gupta, Lu, Levine, & Abbeel, 2015; Levihn & Stilman, 2014) attempt to model tool use via PDDL-like descriptions expressing the applicability and post effects of tool use. In particular, ICARUS (Choi et al., 2018) models bridge or staircase construction via lifted symbolic concepts which can be grounded to real or imagined objects in the environment. The framework presented in this paper takes inspiration from such classic works and builds on the use of learned representations for generalization to new scenes and objects. Further, instead of encoding such knowledge via a symbolic representation for each tool, the framework acquires such knowledge in a data-driven way from human demonstration. Thus, the aforementioned works focus on learning *how* to manipulate a tool. Our paper considers the complementary problem of predicting *which* objects may serve as tools for a given task while delegating the issue of tool manipulation to the works as mentioned earlier.

The works of Fitzgerald et al. (2021) and Gajewski et al. (2019) considers the specific problem of learning the physical motion of tools (such as spatial, hammer, mug, etc.) from kinesthetic demonstration trajectories. Further, the authors present a framework to learn corrections or replacements enabling improvisation when the actual task execution scenario differs from the taught demonstration. The focus of the work is in learning physical motion of tools for short range tasks. This work lifts the problem of dexterous tool manipulation and focuses on utilizing tools to reach goal states (*e.g.*, using a box to transfer multiple objects). This work also extends planning to leverage multiple tools to attain goals (using boxes and later using ramps). Rather than use fine-grained kinesthetic teaching, we instead use demonstrations of.long range plan executions. The two efforts are highly complementary. The tool-use trajectories learned by prior work could potentially be used in our framework to accomplish long-range tasks. Similarly, such works can use the framework presented here to compose skills and generalize to unseen tools.

2.3 Learning Symbolic Action Sequences

Others address the problem of acquiring knowledge for completing high-level task specifications. Puig et al. (2018), Liao, Puig, Boben, Torralba, and Fidler (2019) create a knowledge base of task decompositions as *action sketches* and learn to translate sketches to executable plans. These efforts rely on the causal knowledge of sequences on sub-steps required to achieve an activity which are then contextually grounded. Instead, this work learns compositional tool use required to achieve the task without any causal sequence as input. Huang, Nair, Xu, Zhu, Garg, Fei-Fei, Savarese, and Niebles (2019) learn task decompositions from human demonstration videos. However, their work does not explicitly model the physical constraints of the robot and does not generalize to new environments. Lee et al. (2015) and Huang, Pan, Mulcaire, and Abbeel (2015) learn trajectory-aware

manipulation of deformable objects using a non-rigid registration method and human demonstrations. These methods can effectively handle visual variation in manipulating objects; however, they cannot be extended to generate action sequences to reach goal constraints given an unseen environment. Shridhar, Thomason, Gordon, Bisk, Han, Mottaghi, Zettlemoyer, and Fox (2020) take a similar approach by collecting natural language corpora describing high-level tasks and learn to associate instructions to spatial attention over the scene.

Other works study *relational* planning domains, where model is trained using RL on small problems, but tested zero-shot on large problems (Garg, Bajpai, & Mausam, 2019, 2020). Sharma, Arora, Geißer, Mausam, and Singla (2022b) extend this to imitation learning framework, but these works cannot perform tool generalizations. Boteanu, Kent, Mohseni-Kabir, Rich, and Chernova (2015) present a symbolic system where a robot imitates demonstrations from a single teacher. In new environments, it adapts the plan by performing object replacements using ConceptNet relation edges. A rich set of approaches *learn* affordances by mapping objects to preferred locations or learning the co-use of objects to accomplish a task. For instance, Fitzgerald, Goel, and Thomaz (2018) provide a method of acquiring and transferring such knowledge to new tasks by learning to map between objects in the old and new environments in a task-dependent manner. This work does not explicitly attempt to learn such associations and presents a restricted generalization capacity. Instead, such learning is implicit in the learned policy that predicts a sequence of robot actions to achieve the goal while using tools (e.g., fetching the tool, using the tools and attaining its post effects).

Our approach draws inspiration from the above-mentioned works in that, we learn to predict tools that can be considered as sub-goals to guide planning for a high-level task. In comparison to these approaches, our method provides two key contributions. First, we explicitly model the physical constraints arising from a mobile manipulator interacting in the work space. Second, instead of learning actions predicated on specific object instances, we address generalization to new object instances using primitive spatial and semantic characteristics. We propose a neural model trained using a corpus of multiple and varied demonstrations provided by several teachers. Our model uses a dense embedding of semantic concepts, enabling generalization beyond relationships explicitly stored in ConceptNet.

2.4 Commonsense Knowledge in Instruction Following

Acquisition of common sense knowledge has been previously explored for the task of robot instruction following (Sarathy & Scheutz, 2018). Nyga, Roy, Paul, Park, Pomarlan, Beetz, and Roy (2018) present a symbolic knowledge base for procedural knowledge of tasks that is utilized for interpreting underspecified task instructions. Efforts such as Kho, Hung, and Cunningham (2014) propose a similar database encoding common sense knowledge about object affordances (objects and their common locations). Others such as Jain, Das, Gupta, and Saxena (2015) learn motion preferences implicit in commands. Misra, Sung, Lee, and Saxena (2016) ground instructions for recipe preparation tasks. Their model can generalize to new recipes, but only in environments with previously *seen* objects. In contrast, our model generalizes to worlds with previously *unseen* tools. Others, such as Nair, Srikanth, Erickson, and Chernova (2019a) and (Nair & Chernova, 2020) explore the problem of robot tool construction, *i.e.*, creating tools from parts available in the environment. However, the limited set of commonsense concepts, such as attachment in (Nair et al., 2019a) restricts the space for generalization in such works. A rule-based approach typically does not scale well to develop

a robust generalization model. We thus utilize a commonsense embedding based approach that utilizes ConceptNet vectors to encapsulate the concepts and generalize to unseen tools and object settings.

Chen, Tan, Kuntz, Bansal, and Alterovitz (2020) present an instruction grounding model that leverages common sense taxonomic and affordance knowledge learned from linguistic co-associations. Bisk, Zellers, Bras, Gao, and Choi (2020) consider the problem of learning physical common sense associated with objects and interactions required to achieve tasks from language only data sets. They study this problem in the context of question-answering to enable synthesis of textual responses that capture such physical knowledge. The aforementioned approaches predict latent constraints or affordances for a specified task. This work, additionally predicts the *sequence* of tool interactions implicitly learning the causal relationships between tools use and effects. Specifically, this paper focuses on a learning common sense tool use in the context of following instructions that require multiple object interactions to attain the intended goal.

2.5 Synthetic Interaction Datasets

Virtual environments have been used to collect human demonstrations for high-level tasks. Puig et al. (2018) introduce a knowledge base of actions required to perform activities in a virtual home environment. Shridhar et al. (2020) provide a vision-language dataset translating symbolic actions for a high-level activity to attention masks in ego-centric images. Nyga and Beetz (2018) curated data sets that provide a sequence of *How-To* instructions for tasks such as preparing recipes. Myers et al. (2015) present an affordance detection dataset for tool parts with geometric features. Others such as Jain et al. (2015), Scalise, Li, Admoni, Rosenthal, and Srinivasa (2018) and Mandlekar, Zhu, Garg, Booher, Spero, Tung, Gao, Emmons, Gupta, Orbay, et al. (2018) present simulation environments and data sets for tasks such as learning spatial affordances, situated interaction or learning low-level motor skills. The present data sets possess two limitations that make them less usable for the learning task addressed in this work. First, the data sets are collected using human actors or avatars but do not explicitly model a robot in their environment. Though virtual agents serve as a proxy for the robot, they preclude modeling of the physical constraints and the range of tasks an robot can perform. Second, a majority of the data sets aim at visual navigation and limited physical interaction with objects. They are less amenable to interactions (e.g., containment, pushing and attachment etc.) inherent in tool use. Data sets utilized in robotic tool use literature, including the UMD Part Affordance Dataset (Myers et al., 2015), are confined mainly to local use, such as finding the appropriate tool part for tool use and manipulation. Such data sets are less amenable to multi-stage plans in large workspaces.

3. Problem Formulation

3.1 Robot and Environment Model

We consider a mobile manipulator operating in a known environment populated with objects. An object is associated with a pose, a geometric model and symbolic states such as $\mathrm{Open/Closed}$, $\mathrm{On/Off}$ etc. We consider object relations such as (i) *support* e.g., a block supported on a tray, (ii) *containment*: items placed inside a box/carton and (iii) *attachment*: a nail attached to a wall, and (iv) *contact*: a robot grasping an object. Let s denote the world state that maintains (i) metric information: object poses, and (ii) symbolic information: object states, class type and object relations

as OnTop, Near, Inside and $\mathrm{ConnectedTo}$. Let s_0 denote the initial world state and $\mathcal{O}(\cdot)$ denote a map from world state s to the set of object instances $O = \mathcal{O}(s)$ populating state s. Let τ denote the set of tool objects that the robot can use in its plan. Note that only movable objects in the scene are considered as potential tools. Hence, $\tau \subseteq \mathcal{O}(s)$. Online, the robot may encounter *unseen* objects in its environment.

Let A denote the robot's symbolic action space. An action $a \in A$ is abstracted as $I(o^1, o^2)$, with an action type predicate $I \in \mathcal{I}$ that affects the states of objects $o^1 \in O$ and $o^2 \in O$, for instance, $\mathrm{Move}(\mathrm{fruit}_0, \mathrm{tray}_0)$. Here the arity of the interaction can be 2 or 1 depending on the interaction type. In case of arity of 1, we can drop the second object. Each action in our formulation is realized as a set of object relations in the environment that belongs to $\{\mathrm{OnTop}, \mathrm{Near}, \mathrm{Inside}, \mathrm{ConnectedTo}\}$. The precondition and postconditions of actions are taken directly from prior work (Sharma, Gupta, Tuli, Paul, & Mausam, 2022a). Realization of these conditions using geometric properties is described in Appendix D. We shall also use the notion of a timestamp as a subscript to indicate prediction for each state in the execution sequence. The space of robot interactions include grasping, releasing, pushing, moving an object to another location or inducing discrete state changes (e.g.. opening/closing an object, operating a switch or using a mop). We assume the presence of an underlying low-level metric planner, encapsulated as a robot *skill*, which realizes each symbolic action or returns if the action is infeasible. Robot actions are stochastic, modeling execution errors (unexpected collisions) and unanticipated outcomes (objects falling, changing the symbolic state). Let $\mathcal{T}(\cdot)$ denote the transition function. The successor state s_{t+1} upon taking the action a_t in state s_t is generated from a physics simulator. Let $\eta_t = \{a_0, a_1, \ldots, a_{t-1}\}$ denote the *action history* till time t.

3.2 Semantic Goals and Interactions

The robot's goal is to perform tasks such as transporting or delivering objects to appropriate destinations, making an assembly, clearing or packing items or performing abstract tasks such as illuminating or cleaning the room. The robot is instructed by providing a *declarative* goal g expressing the symbolic constraint between world objects (Hübner, Bordini, & Wooldridge, 2006). For example, the *declarative* goal, "place milk in fridge" can be expressed as a constraint $\mathrm{Inside}(\mathrm{milk}_0, \mathrm{fridge}_0)$ between specific object instances. There may be multiple instance of the same object, for eg. milk carton, in the environment. The sub-index is used to specify which instance is being referred to. Another example is of the task of moving all fruits onto the kitchen table that can be expressed as a list of constraints $\mathrm{OnTop}(\mathrm{apple}_0, \mathrm{table}_0)$, $\mathrm{OnTop}(\mathrm{orange}_0, \mathrm{table}_0)$ and $\mathrm{OnTop}(\mathrm{banana}_0, \mathrm{table}_0)$. Finally, the robot must synthesize a plan to satisfy the goal constraints. Goal-reaching plans may require using some objects as tools, for instance, using a container for moving items, or a ramp negotiate an elevation.

3.3 Predicting Tool Interactions

We assume that the robot is primed with a set of primitive symbolic actions but lacks knowledge about how object characteristics can facilitate their use as in attaining high-level goals. Hence, the robot cannot predict the use of tray-like objects in transportation tasks, or the use of a stick to fetch an object at a distance. An exception is by discovering such characteristics via explicit simulation, which may be infeasible or intractable in large planning domains. Our goal is to learn common sense knowledge about *when* an object can be used as a tool and *how* their use can be sequenced

for goal-reaching plans. We aim at learning a policy π that estimates the next action a_t conditioned on the the goal g and the initial state s (including the action history η_t, such that the robot's *goal-reaching* likelihood is maximized. We adopt the MAXPROB-MDP (Mausam & Kolobov, 2012) formalism and estimate a policy that maximizes the goal-reaching likelihood from the given state. MAXPROB-MDP can be equivalently viewed as an infinite horizon, un-discounted MDP with a zero reward for non-goal states and a positive reward for goal states (Kolobov, Mausam, Weld, & Geffner, 2011). Formally, let $P^{\pi}(s, g)$ denote the *goal-probability* function that represents the likelihood of reaching the goal g from a state s on following π. Let $S_t^{\pi s}$ be a random variable denoting the state resulting from executing the policy π from state s for t time steps. Let $\mathcal{G}(s, g)$ denote the Boolean *goal check* function that determines if the intended goal g is entailed by a world state s as $\mathcal{G}(s, g) \in \{\text{True(T)}, \text{False(F)}\}$. The policy learning objective is formulated as maximizing the likelihood of reaching a goal-satisfying state g from an initial state s_0, denoted as

$$\max_{\pi} P^{\pi}(s_0, g) = \max_{\pi} \sum_{t=1}^{\infty} P\Big(\mathcal{G}(S_t^{\pi s_0}, g) = \text{T} : \mathcal{G}(S_{t'}^{\pi s_0}, g) = \text{F},\ \forall t' \in [1, t)\Big). \tag{1}$$

The policy is modeled as a function $f_\theta(.)$ parameterized by parameters θ that determines the next action for a given world state, the robot's action history and the goal as $a_t = f_\theta(s_t, g, \eta_t)$. We adopt imitation learning approach and learn the function $f_\theta(.)$ from demonstrations by human teachers. Let $\mathcal{D}_{\text{Train}}$ denote the corpus of N goal-reaching plans,

$$\mathcal{D}_{\text{Train}} = \{(s_0^i, g^i, \{s_j^i, a_j^i\}) \mid i \in \{1, \ldots, N\}, j \in \{0, t_i - 1\}\}, \tag{2}$$

where the i^{th} datum consists of the initial state s_0^i, the goal g^i and a state-action sequence

$$\{(s_0^i, a_0^i), \ldots, (s_{t-1}^i, a_{t-1}^i)\}$$

of length t_i. The set of human demonstrations elucidate common sense knowledge about *when* and *how* tools can be used for attaining provided goals. The data set $\mathcal{D}_{\text{Train}}$ supervises an imitation loss between the human demonstrations and the model predictions, resulting in learned parameters θ^*. Online, the robot uses the learned model to sequentially predict actions and execute in the simulation environment till the goal state is attained. We also consider the *open-world* case where the robot may encounter instances of *novel* object categories *unseen* in training, necessitating a *zero-shot* generalization.

4. Technical Approach

We aim at predicting the next robot action a_t, given the world state s_t, the intended goal g and the history η_t of past actions taken the robot. We consider learning to predict multi-step plans requiring complex interaction of objects as tools in possibly novel environments unseen during training. Our technical approach is built on three insights. First, we factor the overall learning task of predicting the action for each environment state by first predicting the tool required to perform the task and then predicting required interaction. This tool conditioned action prediction enables us to decouple the learning problem and make the model training tractable. Second, we incorporate learning dense embeddings of the robot's environment as well as semantic representations for words trained on existing symbolic knowledge corpora. The learned representation allows the robot to generalize its

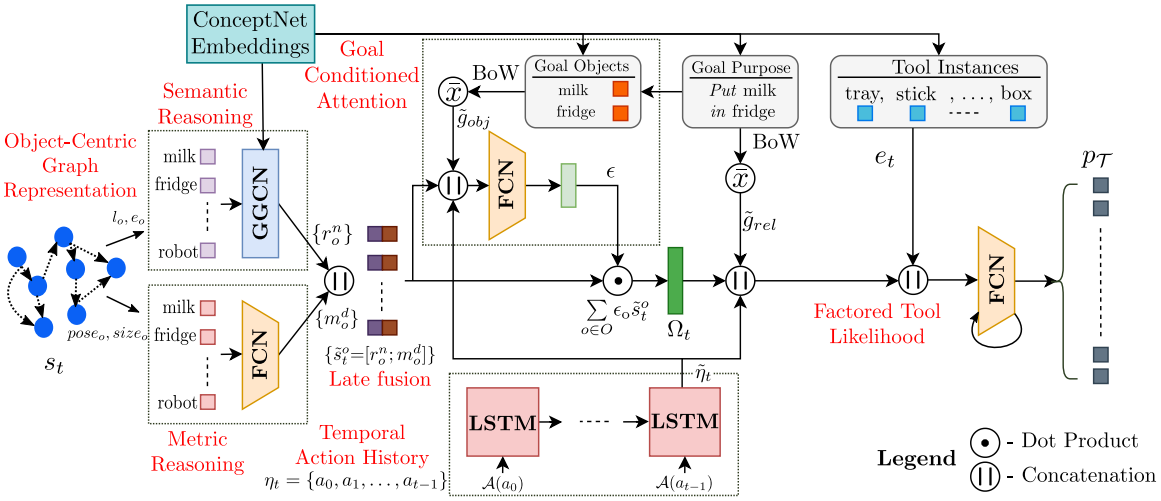


Figure 2: Updated TOOLNET model encodes the metric-semantic world state using graph convolution (GGCN) and fully connected (FCN) layers. The model uses goal information and the robot's action history to attend over a task-specific context, finally predicting the likelihood scores for each tool in the environment. A graph structured representation and inclusion of pre-trained word embeddings (from a knowledge base) facilitate generalization in predicting interactions in novel contexts with new objects unseen in training.

predictions to novel environments populated with novel objects that may share semantic attributes with those encountered during training. Finally, we turn to a corpus of human demonstrated plans to train our learner. The corpus elucidates the commmon sense knowledge of sequencing tools interactions for a task. Formally, we introduce an imitation learner, realized as a hyper-parametric function (a neural network model) denoted as f_θ as follows:

$$a_t = f_\theta\left(s_t, g, \eta_t\right) = f_\theta^{act}\left(f_\theta^{goal}\left(f_\theta^{state}\left(s_t\right), g, f_\theta^{hist}\left(\eta_t\right)\right), f_\theta^{tool}(s_t, g, \eta_t)\right). \tag{3}$$

It adopts an object-centric graph representation, learning a state encoding that fuses metric-semantic information about objects in the environment via function $f_\theta^{state}(\cdot)$. The function $f_\theta^{hist}(\cdot)$ encodes the action history. The model learns to attend over the world state conditioned on the declarative goal and the history of past actions through $f_\theta^{goal}(\cdot)$. We leverage an adapted version of our tool likelihood prediction model TOOLNET, denoted as $f_\theta^{tool}(\cdot)$. Finally, the learned encodings are decoded as the next action for the robot to execute via $f_\theta^{act}(\cdot)$. Crucially, the predicted action is grounded over an *a-priori* unknown state and type of objects in the environment. The predicted action is executed in the environment and the updated state action history is used for estimation at the next time step. Using a neural network as a hyper-parametric function for action prediction enables us to leverage ConceptNet embeddings to generalize and scale with the size of object sets, goal and interaction types. We utilize an imitation learning model with the dataset described in Equation 2 to learn the θ parameters of the neural network. The constituent model components are detailed next.

4.1 Graph Structured World Representation

4.1.1 SEMANTIC REASONING

We first describe the $f_\theta^{state}(\cdot)$ function in equation (3). We denote the robot's current world state s_t as an object-centric graph $G_t = (O, R)$. Each node in the graph represents an object instance $o \in O = \mathcal{O}(s_t)$. The edge set consists of binary relationships OnTop, $\mathrm{ConnectedTo}$, Near and Inside between objects $R \subseteq O \times O$. Let $l_o \in \{0,1\}^p$ represents discrete object states for the object o (e.g. $\mathrm{Open/Closed}$, $\mathrm{On/Off}$). Here, p represents the number of possible states of all objects. Next, it incorporates a pre-trained function $\mathcal{C}(\cdot)$ that embeds a word (like token of an object class or a relation) to a dense distributed representation, such that semantically close tokens appear close in the learned space (Mikolov, Grave, Bojanowski, Puhrsch, & Joulin, 2018). The use of such embeddings enables generalization, which we discuss subsequently. Consider the example goal of "Place the box in the cupboard". In this case we want to satisfy the relationship Inside for cupboard and box. The entire environment can be considered as the graph where relations like $\mathrm{OnTop}(box, table)$ may be true. The pretrained function $\mathcal{C}(\cdot)$ can be any pretrained language model.

Let $e_o = \mathcal{C}(o) \in \mathcal{R}^q$ denote the q-dimensional embedding for an object instance o. The embeddings l_o and e_o model object attributes that initialize the state of each object node in the graph. The local context for each o is incorporated via a Gated Graph Convolution Network (GGCN) (Liao et al., 2019), which performs message passing between 1-hop vertices on the graph. Following (Puig et al., 2018), the gating stage is realized as a Gated Recurrent Unit (GRU) resulting in *graph-to-graph* updates as:

$$\begin{aligned} r_o^0 &= \tanh\left(W_r\left[l_o; e_o\right] + b^r\right), \\ x_o^k &= \sum_{j \in R} \sum_{o' \in N^j(o)} W_j^k r_{o'}^{k-1}, \\ r_o^k &= \mathrm{GRU}\left(r_o^{k-1}, x_o^k\right). \end{aligned} \tag{4}$$

Here, the messages for object o are aggregated over neighbors $N^j(o)$ connected by relation j ($\forall j \in R$) during n convolutions, resulting in an embedding r_o^n for each object instance in the environment. Unlike a fully-connected neural network, a GGCN model facilitates inference over the relations across objects for an input object centric graph. These relations are crucial to predict the relevant actions and achieve the intended goal. GRUs in the convolution iterations facilitate stable training of the model (Li, Tarlow, Brockschmidt, & Zemel, 2015).

4.1.2 FUSING METRIC INFORMATION.

Next, we incorporate the metric information associated with objects. Let $pose_o$ and $size_o$ represent the pose and size/extent (along xyz axes) for each object instance. The pose includes the spatial position and the orientation of the object. We do not explicitly provide the point-cloud geometric model and abstract out the surface information that may be required to infer the utility of objects as tools and instead rely on ConceptNet embeddings for the same. However, in our evaluation (Section 6), we utilize CAD models that closely represent the surface properties to model the physics of tool use. Another point to note here is that for the specific task of utilizing an object as a tool the pose information has little contribution. However, the pose information is required to accurately identify the system state. For instance, the pose information of the door of a fridge indicates whether

the fridge is open or closed. This information is given as a categorical value *i.e.* the state vector l_o to the GGCN model. To explicitly pass this information to the FCN model, we use the pose vector of objects as inputs. The properties are encoded using a d-layer Fully Connected Network (FCN) with a Parameterized ReLU (PReLU) (He, Zhang, Ren, & Sun, 2015) activation as:

$$\begin{aligned} m_o^0 &= \text{PReLU}\left(W_{mtr}^0[pose_o; size_o] + b_{mtr}^0\right) \\ m_o^k &= \text{PReLU}\left(W_{mtr}^k m_o^{k-1} + b_{mtr}^k\right), \end{aligned} \tag{5}$$

resulting in the metric encoding m_o^d for each object in the scene. A world state encoding (for s_t) is obtained by fusing the semantic and metric embeddings as

$$f_\theta^{state}(s_t) = \{\tilde{s}_t^o = [r_o^n; m_o^d] | \; \forall o \in \mathcal{O}(s_t)\}. \tag{6}$$

Late fusion of the two encodings allows downstream predictors to exploit them independently.

4.2 Encoding the Action History

Next, we define the $f_\theta^{hist}(\cdot)$ function in equation (3). The task of deciding the next action is informed by the agent's action history in two ways. First, sequential actions are often temporally correlated. For example, a placing task often involves moving close to the box, opening it and then placing an item inside it. If the previous action involved moving close to a box, this information facilitates the model to leverage localized contextual information and not jump to manipulate another object in the goal specification. This, in tandem with the goal-conditioned attention facilitates seamless action sequence generation. Hence, maintaining the action history can help in prediction of the next action. Second, the set of actions the robot executed in the past provides a local context indicating the objects the the robot may utilize in the future. Formally, we encode the temporal action history η_t using an LSTM. We define action encoding $\mathcal{A}(a_{t-1})$ of $a_{t-1} = I_t(o_{t-1}^1, o_{t-1}^2)$ independent of the object set, as $\mathcal{A}(a_{t-1}) = [\vec{I}_{t-1}; \mathcal{C}(o_{t-1}^1); \mathcal{C}(o_{t-1}^2)]$, where $\vec{I}_{t-1}$ is a one-hot vector over possible interaction types $\mathcal{I}$, and $\mathcal{C}(o_{t-1}^1)$ and $\mathcal{C}(o_{t-1}^2)$ represent the word embeddings of the object instances o_{t-1}^1 and o_{t-1}^2. At each time step t, the LSTM encoder takes in the encoding of previous action, $\mathcal{A}(a_{t-1})$ and outputs the updated encoding $\tilde{\eta}_t$, given as $\tilde{\eta}_t = \text{LSTM}(\mathcal{A}(a_{t-1}), \tilde{\eta}_{t-1})$. This results in the embedding vector

$$f_\theta^{hist}(\eta_t) = \tilde{\eta}_t. \tag{7}$$

4.3 Goal-conditioned Attention

We now define the $f_\theta^{goal}(\cdot)$ function in equation (3). The goal g consists of symbolic relations (e.g. Inside, OnTop etc.) between object instances (e.g., carton and cupboard) that must be true at the end of the robot's plan execution. The declarative goal input to the model is partitioned as relations g_{rel} and the object instances specified in the goal g_{obj}. The resulting encondings are denoted as $\tilde{g}_{rel}$ and $\tilde{g}_{obj}$:

$$\tilde{g}_{rel} = \frac{1}{|g_{rel}|} \sum_{j \in g_{rel}} \mathcal{C}(j) \;\text{ and }\; \tilde{g}_{obj} = \frac{1}{|g_{obj}|} \sum_{o \in g_{obj}} \mathcal{C}(o).$$

Next, the goal encoding and the action history encoding $\tilde{\eta}_t$ is used to learn attention weights over objects in the environment (Bahdanau, Cho, & Bengio, 2015) such that

$$\epsilon_o = \text{softmax}\left(W_g[\tilde{s}_t^o; \tilde{g}_{obj}; \tilde{\eta}_t] + b_g\right), \tag{8}$$

where $\tilde{s}_t^o$ is obtained from $f_\theta^{state}(s_t)$ in (6) and $\tilde{\eta}_t$ is obtained from $f_\theta^{hist}(\eta_t)$ in (7). This results in the attended scene encoding Ω_t as:

$$\Omega_t = f_\theta^{goal}(\tilde{s}_t^o, \tilde{g}_{obj}, \tilde{\eta}_t) = \sum_{o \in O} \epsilon_o \tilde{s}_t^o. \tag{9}$$

The attention mechanism aligns the goal information with the scene learning a task-relevant context, relieving the model from reasoning about objects in the environment unrelated to the task, which may be numerous in realistic environments.

4.4 Tool Likelihood Prediction

Now we define the function that predicts the likelihood scores for each tool in the environment state, *i.e.*, the $f_\theta^{tool}(\cdot)$ function in equation (3). This likelihood score corresponds to the probability that a particular tool could be used to reach a goal state for a given environment state and declarative goal specification. In order to allow the model to generalize to unseen tools, instead of prediction over the pre-defined tool set $\hat{\tau}$, we allow the model to predict a likelihood score of a tool t (which may not be present in any of the scenes in the training set) to be used as a tool using the object embedding ($e_t = \mathcal{C}(t)$ as described in Section 4.1.1). This recurrence is shown in the factored tool likelihood module in Figure 2. The prediction is made using the encoding of the state, i.e the attended scene embedding Ω_t, the relational description of the goal $\tilde{g}_{rel}$ and the action history encoding $\tilde{\eta}_t$. Likelihood of each tool is computed for each t using,

$$p_t = \text{sigmoid}(W[\Omega_t; \tilde{g}_{rel}; \tilde{\eta}_t; e_t]) \; \forall \; t \in \hat{\tau}, \tag{10}$$

where Ω_t is obtained using (9) and $\tilde{\eta}_t$ is obtained using (7). We now predict over a tool set τ, which may be different from the fixed toolset seen during training set. We include the possibility of having tools present at inference time that are unseen during training. The factored style likelihood prediction enables our model to be flexible to make likelihood predictions for unseen tools. We now use this to define the likelihood score for each object as p_t for tools and 0 otherwise. Here p_t is the probability that tool t can be used to complete the goal. Specifically,

$$f_\theta^{tool}(s_t, g_t, \eta_t) = \{p_o \text{ if } o \in \tau \text{ else } 0 \mid \forall o \in \mathcal{O}(s_t)\}. \tag{11}$$

Unlike the original TOOLNET model (Bansal et al., 2020), $f_\theta^{tool}(\cdot)$ function in this work can predict the tool likelihood scores for each time-step t and utilizes action history encoding $\tilde{\eta}_t$ to capture temporal context.

4.5 Robot Action Prediction

We now take the encoded information about the world state, goal and action history to decode the next symbolic action $a_t = I_t(o_t^1, o_t^2)$. The three components I_t, o_t^1 and o_t^2 are predicted auto-regressively, where the prediction of the interaction, I_t is used for the prediction of the first object, o_t^1 and both their predictions are used for the second object prediction, o_t^2. For the object predictors o_t^1 and o_t^2, instead of predicting over a predefined set of objects, our model predicts a likelihood score of each object $o \in O$ based on its object embedding $\tilde{s}_t^o$ and tool likelihood score p_o from (11). It then selects the object with highest likelihood score. The resulting factored likelihood allows the

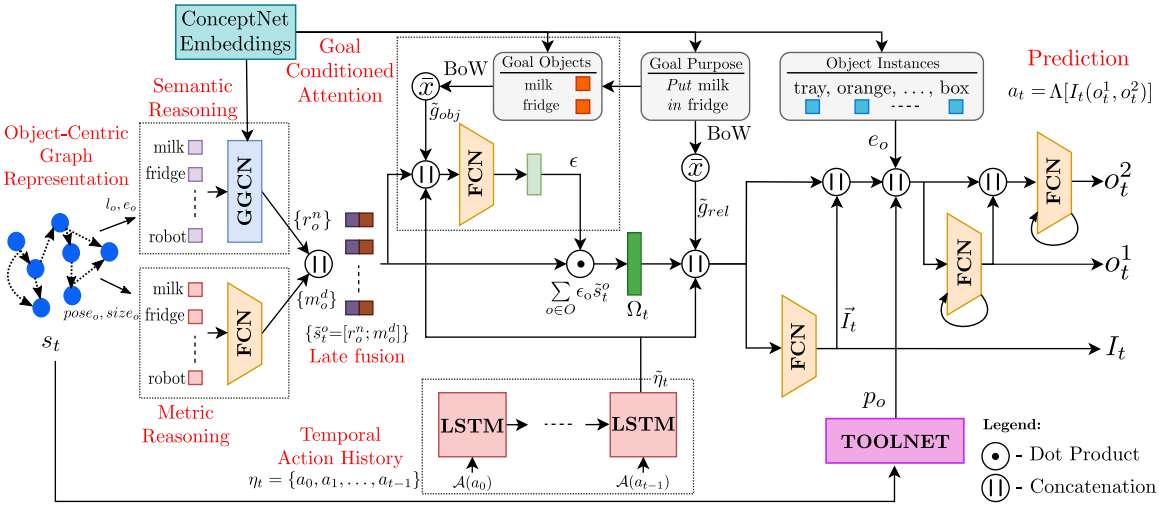


Figure 3: TOOLTANGO neural model is architecturally similar to TOOLNET but also incorporates tool likelihood scores predicted from the TOOLNET model, finally decoding the next symbolic action for the robot to execute. The interaction type and objects are predicted auto-regressively and in a factored style.

model to generalize to an *a-priori* unknown number and types of object instances:

$$I_t = \text{argmax}_{I \in \mathcal{I}} \left(\text{softmax}(W_I[\Omega_t; \tilde{g}_{rel}; \tilde{\eta}_t] + b_I)\right), \tag{12}$$

$$\begin{aligned} o_t^1 &= \text{argmax}_{o \in O} \alpha_t^o \\ &= \text{argmax}_{o \in O} \left(\sigma(W_\alpha[\Omega_t; \tilde{g}_{rel}; \tilde{\eta}_t; e_o; p_o; \vec{I}_t] + b_\alpha)\right), \end{aligned} \tag{13}$$

$$o_t^2 = \text{argmax}_{o \in O} \left(\sigma(W_\beta[\Omega_t; \tilde{g}_{rel}; \tilde{\eta}_t; e_o; p_o; \vec{I}_t; \alpha_t^o] + b_\beta)\right). \tag{14}$$

Here α_t^o denotes the likelihood prediction of the first object. Finally, we impose grammar constraints (denoted as Λ) at inference time based on the number of arguments that the predicted interaction I_t accepts. If I_t accepts only one argument, only o_t^1 is selected, otherwise both are used. Thus, predicted action for time-step t is denoted as

$$a_t = f_\theta^{act}(\Omega_t, \tilde{g}_{rel}, \tilde{\eta}_t) = \Lambda[I_t(o_t^1, o_t^2)]. \tag{15}$$

This action is then executed by the robot in simulation. Our simulation model follows closely to the one developed by Puig et al. (2018) wherein symbolic actions such as moveTo, pick and place are realized as relations in the environment state using a PyBullet simulator. For instance, the moveTo action changes the xyz coordinates of the robot such that it is Near the target object. Similarly, pick establishes a connectedTo relation between the target object and robot arm. For implementation specific details, visit `https://github.com/reail-iitd/tango/wiki`. The executed action and resulting world state is provided as input to the model for predicting the action at the next time step. The $f_\theta^{act}(\cdot)$ function denotes the TOOLTANGO model. In our original TANGO model presented in (Tuli et al., 2021), the p_o tool likelihood score was not part of equations (13) and (14).

4.6 Word Embeddings Informed by a Knowledge Base

TOOLTANGO uses word embedding function $\mathcal{C}(\cdot)$ that provides a dense vector representation for word tokens associated with object class and relation types. Contemporary models use word em-

beddings acquired from language modeling tasks (Mikolov et al., 2018). We adopt embeddings that are additionally informed by an existing knowledge graph $\mathrm{ConceptNet}$ (Speer et al., 2017) that provides a sufficiently large knowledge graph connecting words with edges expressing relationships such as $\mathrm{SimilarTo}$, IsA, $\mathrm{UsedFor}$, PartOf and $\mathrm{CapableOf}$. Word embeddings (Mikolov et al., 2018) can be *retro-fitted* such that words related using knowledge graph embeddings are also close in the embedding space (Speer, Chin, & Havasi, 2019). Using such (pre-trained) embeddings incorporates *general purpose* relational knowledge to facilitate richer generalization for downstream policy learning. Thus, given an object, say $apple$, the word embedding function $\mathcal{C}(.)$ returns a dense $\mathrm{ConceptNet}$ vector that corresponds to this token. The complete sequence of steps is summarized in Figure 3.

4.7 Model Training

We decompose the action prediction task into first predicting tool, *i.e.*, the $f_\theta^{tool}(\cdot)$ function and then evaluating the $f_\theta^{act}(\cdot)$ function. We also decompose model training. First, we train the tool prediction model. The loss used to train this model is Binary Cross-Entropy with each $t \in \hat{\tau}$ acting as a class. The class label, y_t^i is assigned 1 if it is used in a given demonstration, i and 0 otherwise. We also use categorical weights based on plan execution time to encourage shorter plans (Bansal et al., 2020). However, the knowledge of the time taken for different plans has not been injected into the model. In order to make this notion explicit to the model we use loss weighting such that for the dataset $\mathcal{D}_{\mathrm{Train}}$ defined in (2),

$$\mathcal{L} = -\sum_i \alpha_i \sum_j \sum_{t \in \tau} y_{j,t}^i \log(p_{j,t}^i) + (1 - y_{j,t}^i) \log(1 - p_{j,t}^i), \tag{16}$$

where, $p_{j,t}^i$ is obtained from $f_\theta^{tool}(s_j^i, g^i, \eta_j^i)$ for each datapoint in $\mathcal{D}_{\mathrm{Train}}$ and α_i is a multiplier that is high for optimal plans (shortest among human demonstrations) and low otherwise.

For a trained tool prediction model, we then train the $f_\theta^{act}(\cdot)$ TOOLTANGO model, fine-tuning the weights of our updated TOOLNET function $f_\theta^{tool}(\cdot)$. Here too we use the Binary Cross-Entropy loss, with the loss for the three predictors (action and the two objects) being added independently. Additional model training and hyperparameter details are given in Appendix B.

This decomposed training style has direct implications on the training stability. First training TOOLNET to predict the likelihood scores for tools enables action prediction informed by the likelihood scores of each tool in the environment.

5. Data Collection Platform and Annotation

5.1 Data Collection Environment

We develop a low-fidelity simulation environment where the robot can take actions and interact with objects. The low-level physics of motion is less important as we are primarily concerned with the symbolic effects on the world state. To do this, we use PyBullet, a physics simulator (Coumans & Bai, 2016), to generate home and factory-like environments populated with a virtual mobile manipulator (a Universal Robotics (UR5) arm mounted on a Clearpath Husky mobile base). The world state is object-centric including the metric locations of objects and the discrete states of symbolic attributes and relations. The objects in the domains were derived from real-world home and factory scenes that span Facebook Replica Dataset (Straub, Whelan, Ma, Chen, Wijmans, Green, Engel,

Domain	Plan lengths	Objects interacted with in a plan	Tools used in a plan	Sample objects	Sample goal specifications
Home	23.25±12.65	4.12±1.97	0.93±0.70	floor[1], wall, fridge[123], cupboard[123], tables[1], couch[1], **big-tray**[1], **tray**[1], **book**[1], paper, cubes, light switch[4], bottle, **box**[2], fruits, **chair**[15], **stick**, dumpster[2], milk carton, shelf[1], **glue**[6], **tape**[6], **stool**[15], **mop**[8], **sponge**[8], **vacuum**[8], dirt[7], door[2]	1. Place milk in fridge, 2. Place fruits in cupboard, 3. Remove dirt from floor, 4. Stick paper to wall, 5. Put cubes in box, 6. Place bottles in dumpster, 7. Place a weight on paper, 8. Illuminate the room.
Factory	38.77±23.17	4.38±1.85	1.44±0.97	floor[1], wall, **ramp**, worktable[1], **tray**[1], **box**[2], crates[1], **stick**, long-shelf[1], **lift**[1], cupboard[123], **drill**[4], **hammer**[49], **ladder**[5], **trolley**[2], **brick**, **blow dryer**[48], **spraypaint**[4], **welder**[4], generator[4], **gasoline**, **coal**, **toolbox**[2], **wood cutter**[4], **3D printer**[4], screw[9], nail[9], **screwdriver**[49], wood, platform[1], oil[7], water[7], board, **mop**[8], **glue**[6], **tape**[6], **stool**[15]	1. Stack crated on platform, 2. Stick paper to wall, 3. Fix board on wall, 4. Turn on the generator, 5. Assemble and paint parts, 6. Move tools to workbench, 7. Clean spilled water, 8. Clean spilled oil.

Table 1: Dataset characteristics. The average plan length measured as the length of action sequence), number of objects interacted in plan and number of tools used in plans with object and goal sets for Home and Factory domains. Object positions were sampled using Gaussian distribution. Objects in bold can be used as tools. Legend:- [1]: surface, [2]: can open/close, [3]: container, [4]: can operate, [5]: can climb, [6]: can apply, [7]: can be cleaned, [8]: cleaning agent, [9]: can 3D print. Objects in bold can be used as tools. Object affordances derived from properties in the ConceptNet graph (Speer et al., 2017). Stool/ladder are objects used to represent a tool for raising the height of the robot.

Robot Actions
Push, Climb up/down, Open/Close, Switch on/off, Drop, Pick, Move to, Operate device, Clean, Release material on surface, Push until force
Object Attributes
Grabbed/Free, Outside/Inside, On/Off, Open/Close, Sticky/Not Sticky, Dirty/Clean, Welded/Not Welded, Drilled/Not Drilled, Driven/Not Driven, Cut/Not Cut, Painted/Not Painted
Semantic Relations
On top, Inside, Connected to, Near
Metric Properties
Position, Orientation, Size

Table 2: Domain Representation. Robot symbolic actions, semantic attributes, relations to describe the world state and objects populating the scene in Home and Factory Domains.

Mur-Artal, Ren, Verma, et al., 2019). These scenes were made to look as photo realistic as possible. The objects on other hand were chosen to span the YCB dataset (Calli, Singh, Bruce, Walsman, Konolige, Srinavasa, Abbeel, & Dollar, 2017). The CAD model for each object obtained from open-source repositories such as the Google 3D Warehouse[1]. This was done in order to keep the simulated physics as real as possible. The set of objects in the two domains are listed in Table 1.

Each object in the environment is associated with a metric location and physical extent and may optionally possess discrete states such as $\mathrm{Open/Closed}$, $\mathrm{On/Off}$, etc. The world model is assumed to possess spatial notions such as Near or Far. Objects in the world model can be supported by, contained within or connected with other objects (or the agent). Hence, we include semantic relations such as OnTop, Inside, $\mathrm{ConnectedTo}$ etc. More details in Appendix D.

1. https://3dwarehouse.sketchup.com/

The robot possesses a set of behaviours or symbolic actions such as Moving towards an object, Grasping, Releasing/Dropping or Pushing an object or Operating an entity to imply actions that induce discrete state changes such as opening the door before exiting, turning on a switch etc. We assume that the robot's actions can be realized by the presence of an underlying controller. We encode the geometric requirements for actions as symbolic pre-conditions. Examples include releasing an object from the gripper before grasping another, opening the door before trying to exit the room. The set of possible actions and the range of interactions are listed in Table 2.

The robot is tasked with goals that involve multiple interactions with objects derived from standardized data sets (Calli et al., 2017). These goals include: (a) *transporting* objects from one region to another (including space on top of or inside other objects), (b) *fetching* objects, which the robot must reach, grasp and return with, and (c) *inducing state changes* such as illuminating the room or removing dirt from floor. We assume that the robot is instructed by providing *declarative* goals. For example, the task of moving all fruits on top of the kitchen table can be modeled as a set intended constraints among the interacting objects. The effects of actions such as pushing or moving are simulated via a motion planner and propagated to the next time step. Abstract actions such as attachment, operating a tool or grasping/releasing objects are encoded symbolically as the establishment or release constraints. The simulation for these actions is coarse and considers their symbolic effects forgoing the exact motion/skills needed to implement them. We assume that the robot can realize abstract actions through low-level routines.

We present the human agents with a scene and goal conditions we want to attain and receive actions in a custom grammar. The instructions are grounded in the world and the agent and the virtual environment show its effects to input the next action. This is repeated till a goal state is reached. Figure 4 illustrates the interactive platform used for data collection. Using a curated dataset, the robot must synthesize a plan of executable actions to satisfy the goal constraints. The presence of a rich space of interactions gives rise to plans with multiple interactions between objects. For example, "packing items into a basket and carrying the basket to the goal region", "using a stick to fetch and drop an object beyond reach into a box", "using a ramp/stool to elevate itself to fetch an object".

5.2 Annotated Corpus

To curate an imitation learning dataset, we recruit human instructors and provide them with goals. They instruct the robot by specifying a sequence of symbolic actions (one at a time) to achieve each goal. Each action is simulated so that they can observe its effects and the new world state. We encourage the instructors to to complete the task as quickly as possible, making use of available tools in the environment. To familiarize them with the simulation platform, we conduct tutorial sessions before data collection. Our resulting dataset consists of diverse plans with different action sets and object interactions. We collected plan traces from 12 human subjects using domain randomization with 10 scenes and 8 semantic goals resulting in a corpus of 708 and 784 plans for the home and factory domains. Figure 5a and 5b show number of interactions with 10 most interacted objects and frequency of 10 most frequent actions respectively. The complete set of objects and goals is given in Table 1. We also perform data augmentation by perturbing the metric states in the human plan trace, performing random replacements of scene objects and validating plan feasibility in simulation. The process results in 3540 and 3920 plans, respectively. Variation was observed in tool use for an instructor for different goals, and within similar goals based on context. The annotated corpus was

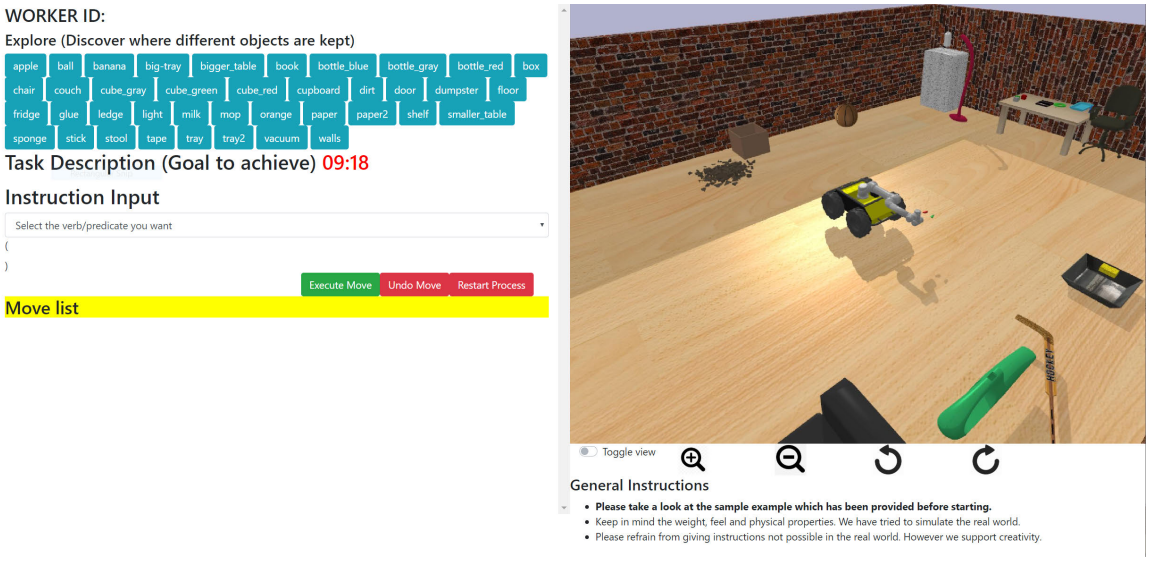
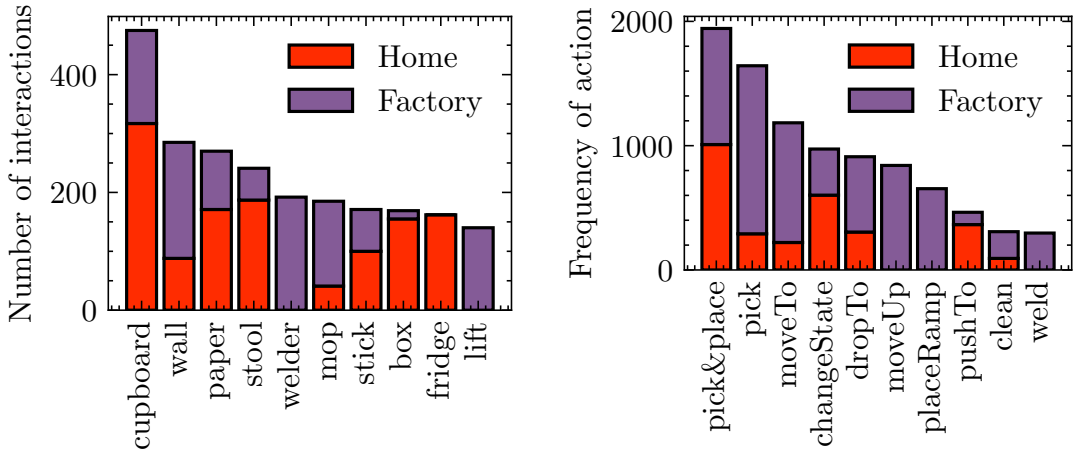


Figure 4: Data Collection Interface. The human teacher instructs (left) a virtual mobile manipulator robot by specifying symbolic actions. The human-instructed plan is simulated and visualized (right). The user is shown the goal needed to be completed in text form. They select the user interaction, first object and second object to instruct the robot. The interface can be seen in action in the `https://www.youtube.com/watch?v=lUWU3rK1Gno`.



(a) Object interactions

(b) Symbolic actions

Figure 5: Data set Characteristics. Distribution of plans with plan length for home and factory domains. Frequency of interaction of top 10 objects and frequency of actions for top 10 actions. The collected data set contains diverse interactions in complex spaces.

split as $(75\% : 25\%)$ forming the Training data set and the Test data set to evaluate model accuracy. A 10% fraction of the training data was used as the Validation set for hyper-parameter search. No data augmentation was performed on the Test set.

5.3 Generalization Test Set

In order to asses the model's capacity to generalize to unseen worlds, we curate a second test environment populated by instances of novel object types placed at randomized locations. The following sampling strategies were used: (i) *Position*: perturbing and exchanging object positions in a scene. (ii) *Alternate*: removal of the most frequently used tool in demonstrated plans evaluating the ability to predict the alternative, next best tool to use. (iii) *Unseen*: replacing an object with a similar object, which is not present in training. (iv) *Random*: replacing a tool with a randomly picked object which is *unrelated* to the task. (v) *Goal*: replacing the goal objects (objects included as part of the goal specifications). For example: milk and fridge in the goal "put milk inside fridge". Here milk may be replaced with another similar object, such as apple and fridge by a container, such as cupboard. This process resulted in a Generalization Test set with 7460 (goal, plan) pairs.

6. Experiments

We first present the results corresponding to high-level tool prediction that leverages an updated TOOLNET model. We then present the goal-reaching performance of TANGO and the unified TOOLTANGO model. Finally, we present a detailed analysis of the resulting plans in terms of generalization, robustness to execution errors and plan efficiency.

6.1 Sequential Tool Prediction

6.1.1 BASELINE

We compare against the basic GGCN model of Section 4.1.1 as our baseline model. This model is similar to the *ResActGraph* model proposed by Liao et al. (2019) and its encoder incorporates technical ideas from recent imitation learning works (Shridhar et al., 2020) on action prediction. In the subsequent discussion, we consider similar size of the hyperparameter set across all models performing the same task.

6.1.2 EVALUATION METRICS

We first compare the accuracy of the updated TOOLNET model on the test and generalization test sets. We test TOOLNET's tool prediction capabilities in two settings. In the first setting, we use the dataset as described in Section 5 and split it according to the scene instance, *i.e.*, home and factory. We use accuracy as our performance measure. Here, a tool prediction is deemed correct if the predicted tool is used in at least one of the various annotated plans for the $(\text{goal}, \text{scene})$ pair and incorrect otherwise.

6.1.3 RESULTS

Table 3 shows the final accuracies on the test-set and generalization test-set with individual accuracies for each test-type. On the regular test set, TOOLNET outperforms the baseline by 47.87 (149% higher) and 54.23 (259% higher) accuracy points on Home and Factory domains, respectively.

A similar pattern is found in the generalization test set, where each improvement is 58.94 (278% higher) and 58.51 (300% higher) accuracy points for Home and Factory domains. The reasoning behind the improvements is as follows. The goal-conditioned attention gives major improvement in the *Goal* generalization test type, since goal objects get replaced in those examples. Explicitly

Model	Test Set		Generalization		Generalization Test cases				
	Home	Factory	Home	Factory	Position	Alternate	Unseen	Random	Goal
Baseline (GGCN)	32.01	20.89	21.22	19.50	8.73	6.29	12.22	9.81	46.31
Updated TOOLNET	**79.87**	**75.12**	**80.16**	**78.01**	**75.55**	**58.83**	**60.74**	**49.97**	**94.10**

Table 3: A comparison of the updated TOOLNET model with its previous version and GGCN baseline for the accuracy of prediction of tool sequences instead of single tool. Highest scores are shown in bold.

biasing the model to use the features of those objects (through conditioned attention) increases their importance, and likely reduces overfitting. An example for *Alternate* case is when *generator* is specified in the goal, and wood (the fuel for generator, and the most likely tool) is made absent from the scene. The model could err in giving attention to the *wood-cutter* tool, which is often correlated with wood. However, conditioned attention gives low attention to *wood-cutter* and predicts *gasoline*, instead.

Moreover, factored likelihood predictions helps the most in *Unseen* cases, since without this component, the model cannot predict any unseen tool. A decent performance of earlier models in this case is attributed to alternative possible correct answers (any alternative seen tool or no-tool) due to multiple annotations per scenario. The ConceptNet embeddings likely contain commonsense knowledge about unseen tools and objects, for example, whether a new tool is flat or not (which should help in ascertaining whether it can be used for transport or not). Using these embeddings makes huge improvement in *Unseen* cases where entirely new objects are to be predicted as tools. Finally, giving higher weight to optimal plans allows the model to differentiate tools by plan execution time and not human usage frequency. This helps in improved metric generalization, predicting nearby tools in the *Position* test case. Overall, the complete architecture provides the maximum generalization accuracy among all models.

6.2 Action Prediction

6.2.1 BASELINES

We compare to the following three baseline models. (1) *ResActGraph* model (Liao et al., 2019), augmented with FastText embeddings (Mikolov et al., 2018). (2) *Affordance-only* baseline inspired from (Hermans, Rehg, & Bobick, 2011) that learns a co-association between tasks and tools, implemented by excluding the graph convolutions and attention from TANGO. (3) *Vanilla Deep Q-Learning (DQN)* approach (Bae, Kim, Kim, Qian, & Lee, 2019) that learns purely by interactions with a simulator, receiving positive reward for reaching a goal state.

6.2.2 TOOL PREDICTION ACCURACY

Our experiments use the following accuracy metrics for model evaluation: (i) *Action prediction accuracy*: the fraction of tool interactions predicted by the model that matched the human demonstrated action a_t for a given state s_t, and (ii) *Plan execution accuracy*: the fraction of estimated plans that are successful, i.e., can be executed by the robot in simulation and attain the intended goal (in max. 50 steps).

Model	Action Prediction		Plan Execution		Generalization Plan Execution Accuracy						
	Home	Factory	Home	Factory	Home	Factory	Position	Alternate	Unseen	Random	Goal
Baseline (ResActGraph)	27.67	45.81	26.15	0.00	12.38	0.00	0.00	0.00	0.00	25.10	9.12
Affordance Only	46.22	52.71	52.12	20.39	44.10	4.82	17.84	47.33	29.31	29.57	34.85
DQN	-	-	24.82	17.77	15.26	2.23	0.00	0.00	12.75	9.67	4.21
TANGO	59.43	60.22	92.31	71.42	**91.30**	**60.49**	**93.44**	**77.47**	**81.60**	59.68	**59.41**
Model Ablations											
- GGCN (World Representation)	59.43	60.59	84.61	27.27	78.02	38.70	70.42	58.79	60.00	56.35	38.64
- Metric (World Representation)	58.8	60.84	84.61	62.34	72.42	51.83	59.68	67.19	60.79	**84.47**	21.70
- Goal-Conditioned Attn	53.14	60.35	53.85	11.69	37.02	8.80	35.33	15.05	32.14	41.67	6.51
- Temporal Action History	45.91	49.94	24.61	0.00	8.55	0.00	0.00	0.00	0.00	30.56	1.15
- Factored Likelihood	61.32	**61.34**	**95.38**	**85.71**	34.22	43.44	90.50	14.82	30.65	64.67	53.26
- ConceptNet	**63.52**	60.35	89.23	57.14	81.86	56.97	82.33	68.61	74.57	65.73	47.92
- Constraints	57.23	57.74	64.62	37.66	62.98	41.95	84.95	45.39	39.99	36.11	84.85
- Auto-regression	56.60	60.22	69.23	50.65	73.75	53.32	71.24	66.43	61.27	70.51	34.66

Table 4: A comparison of *Action prediction* and *Plan execution* accuracies for the baseline, the proposed TANGO model, and ablations. Results are presented for test and generalization data sets (under five sampling strategies) derived from the home and factory domains. *Accuracy Prediction* is the percentage of predicted actions matching the human input on the Test set. *Plan Execution* is the percentage of plans successfully executed in the Test set. *Generalization Plan Accuracy* is the percentage of plans successfully executed in the set. Highest scores shown in bold.

We first present the results for the vanilla TANGO model and later present improvements with the TOOLTANGO.

6.2.3 COMPARING TANGO WITH BASELINES

Table 4 (top half) compares the TANGO model performance with the baseline models. The TANGO model shows a $14-23$ point increase in *Action prediction accuracy* and a $66-71$ points increase in the *Plan execution accuracy* when compared to the *ResActGraph* baseline. Note that the *ResActGraph* model learns a scene representation assuming a fixed and known set of object types and hence can only generalize to new randomized scenes of known objects. In contrast, the TANGO model can not only generalize to randomized scenes with known object types (sharing the GGCN backbone with *ResActGraph*) but can to novel scenes new object types (relying on dense semantic embeddings) and an a-priori unknown number of instances (enabled by a factored likelihood).

The *Affordance-only* baseline model is confined to learning the possible association between a tool object type and the task specified by the human (largely ignoring the environment context). This approach addresses only a part of our problem as it ignores the sequential decision making aspect, where tools may need to be used in sequence to attain a goal. Finally, the vanilla DQN baseline achieves less than 20% policy accuracy (even after a week of training). In contrast, the TANGO model shows accurate results after training on imitation data for $12-18$ hours. The challenges in scaling can be attributed to the problem size ($\approx$1000 actions), long plans, sparse and delayed rewards (no reward until goal attainment).

Next, we assess the zero-shot transfer setting, i.e., whether the model can perform common sense generalization in worlds with new objects unseen in training. The same table shows that the plans predicted by TANGO lead to an increase of up to 56 points in plan execution accuracy on Generalization Test set over the best-performing baseline model. This demonstrates accurate

prediction and use of unseen tool objects for a given goal. Specifically, in the home domain, if the *stool* is not present in the scene, the model is able to use a *stick* instead to fetch far-away objects. Similarly, if the robot can predict the use of a box for transporting objects even if it has only seen the use of a tray for moving objects during training. The *ResActGraph* model is unable to adapt to novel worlds and obtains zero points in several generalization tests.

The poorer performance of the *Affordance-only* model can again be attributed to the fact that planning tool interactions involves sequential decision-making. Even if the robot can use affordance similarity to replace a *tray* object with a *box*, it still needs to predict the opening of the *box* before placing an item in its plan for a successful execution. This is corroborated by the drop in performance for the *Unseen* generalization tests for this model by 52.3 points. Finally, the vanilla DQN model lacks a clear mechanism for transferring to novel settings, hence shows poor generalization in our experiments.

6.2.4 ABLATION ANALYSIS OF TANGO COMPONENTS

We analyze the importance of each component of the TANGO model by performing an ablation study. Table 4 (lower half) presents the results. For a fair comparison, the model capacities remain the same during the ablation experiments.

The model builds on the *GGCN* environment representation encoding the inter-object and agent-object relational properties. The ablation of the GGCN component results in a reduction of 22% in the generalization accuracy in the factory domain (where tools may be placed at multiple levels in the factory). The inclusion of this component allows the robot to leverage relational properties such as OnTop to predict the use of tools such as a ramp to negotiate an elevation or a stick to fetch an object immediately beyond the manipulator's reach.

The *Metric* component encodes the metric properties of objects in the scene such as positions, size etc. Experiments demonstrate its effectiveness in prediction tool interactions based on relative sizes of interacting objects. E.g., the model successfully predicts that *fruits* can be transported using a *tray* but larger *cartons* require a *box* for the same task. The ablation of this component leads to a reduction of 10.2 points in the *Alternate* generalization tests as the ablated model unable to adapt the tool when there are unseen objects with different sizes than those seen during training.

Next, we assess the impact of removing the *Goal-Conditioned Attention component*. This experiment shows a a significant reduction (≈ 50 points) in the *Plan execution accuracy* on the Generalization Test set, particularly in scenes with a large number of objects. The attention mechanism allows learning of a restricted context of tool objects that may be useful for attaining the provided goal, in essence, filtering away goal-irrelevant objects populating the scene. Additionally, note that the inclusion of this component allows tool predictions to be *goal-aware*. Consequently, we observe ablating this component leads to a reduction of 53 points in the *Goal* generalization test set where the goal objects are perturbed.

The *Action History* component utilizes the agent's past interactions for the purpose of predicting the next tool interaction. The inclusion of this component allows learning of correlated and commonly repeated action sequences. For instance, the task of exiting from a room typically involves a plan fragment that includes moving to a door, opening it and exiting from the door and are commonly observed in a number of longer plans. The ablation of this component leads to erroneous predictions where a particular action in a common plan fragment is missing or incorrectly predicted. E.g., a robot attempting to pick an object inside an enclosure without opening the lid. In our ex-

Model	Action Prediction		Plan Execution		Generalization Plan Execution Accuracy						
	Home	Factory	Home	Factory	Home	Factory	Position	Alternate	Unseen	Random	Goal
TANGO	59.43	60.22	92.31	71.42	91.30	60.49	93.44	77.47	81.60	59.68	59.41
TOOLTANGO	**64.12**	**63.72**	**95.69**	**77.01**	**92.88**	**62.97**	**94.38**	**88.12**	**92.71**	**78.01**	**64.43**

Table 5: A comparison of TANGO with TOOLTANGO model.

periments, we observe that ablating the model leads to a significant decrease in goal reach-ability, causing a 70 point decrease in the *Plan Execution accuracy* and 72 point drop in the *Generalization accuracy*.

The need for generalization to novel scenes implies that our model cannot assume a fixed and a-priori known set of objects that the robot can interact with. Generalization to an arbitrary number of objects in the scene is accomplished by factoring model predictions over individual objects in a recurrent manner. Ablating the factored likelihood components results in a simpler model that performs predictions over a known fixed-size object set. The simplified model displays a higher action-prediction and plan-execution accuracies in known world. Crucially, we observe that ablating this component results in a significant decrease of 51 and 63 points in the *Unseen* and the *Alternate* generalization test sets.

Finally, the *ConceptNet embeddings* are important for semantic generalization to unseen tool types. We replace ConceptNet embeddings with FastText (Mikolov et al., 2018) embeddings in the *-ConceptNet* model to show their importance. The *-ConceptNet* model shows poorer generalization (6.5% decrease) as it models word affinity as expressed in language only. ConceptNet embedding space models relational affinity between objects as present in the knowledge-base.

6.2.5 COMPARING TANGO WITH TOOLTANGO

Table 5 compares the performance of TOOLTANGO with the TANGO model. TOOLTANGO gives improved scores in both domains and for every generalization test case. The *Action prediction* accuracy of TOOLTANGO is higher than TANGO by 4.69 and 3.50 points for Home and Factory domains, respectively. Similarly, we see an increase in the *Plan execution accuracy* both for Test and Generalization Test sets. We get 3.38- 5.59 higher points on Test and 1.58-2.48 points in Generalization Test set for Home and Factory domains. The improvement is higher in Factory domain where the probability of using multiple tools in the same plan to reach a goal state is higher. This is particularly due to the independent tool prediction in TOOLTANGO that aids action prediction in complex settings requiring longer action sequences (more details in Section 6.3). The performance improvement is highest for the *Alternate* and *Random* cases where the tools are replaced by an alternate tool or another non-tool object. This shows the importance of independent tool-likelihood score prediction in the TOOLTANGO model.

6.3 Analysis of Resulting Plans

6.3.1 EVIDENCE OF GENERALIZATION

Figure 6 shows the robot using the learned model to synthesize a plan for a declarative goal. Here, if the goal is to transport fruits and human demonstrates usage of *tray* and the model never sees *box* while training, TOOLTANGO uses *box* in a scene where tray is absent, showing that it is able to

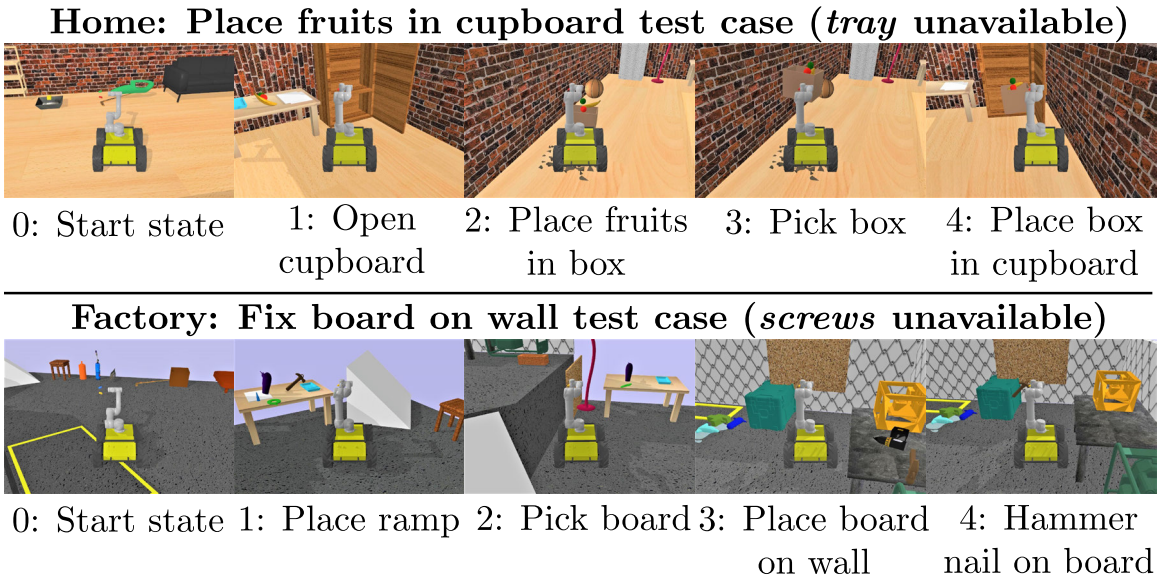


Figure 6: A simulated robot manipulator uses TOOLTANGO to synthesize tool interactions in novel contexts with unseen objects. (top) TOOLTANGO predicts the use of a *box* when *tray* is unavailable. (bottom) TOOLTANGO predicts the use of *hammer* and *nails* when *screws* are unavailable.

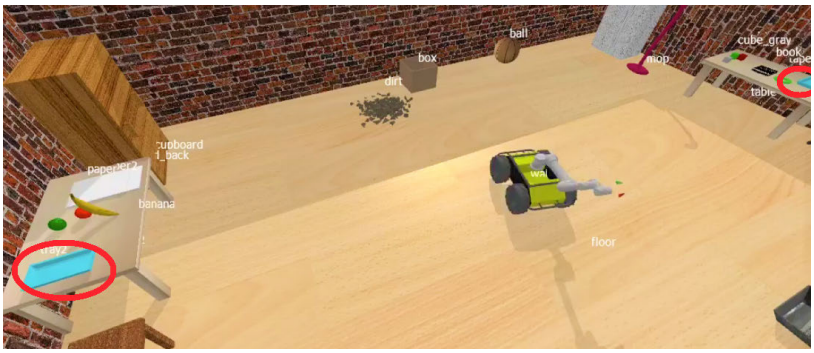


Figure 7: The model predicts the instance of *tray* (on the left) which is closer to the *fruits* (goal objects) other instance (one on the right).

predict semantically similar tools for task completion. Similarly, for the goal of fixing a board on the wall, if humans use *screws* the agent uses *nails* and *hammer* when screws are absent from the scene. Figure 7 shows how the model uses the position information of tool objects to predict the tool closer to the goal object or the agent. The world representation encodes the metric properties of objects (position and orientation) that allows the robot to interact with nearer tool objects.

6.3.2 ROBUSTNESS TO ERRORS

Figure 8 shows the robustness to unexpected errors and stochasticity in action execution. Consider the task of "fetching a carton", where the milk carton is on an elevated platform, the model predicts the uses a stool to elevate itself. The carton falls due to errors during action execution. Following which, the robot infers that the stool is no longer required and directly fetches the carton. Similarly,

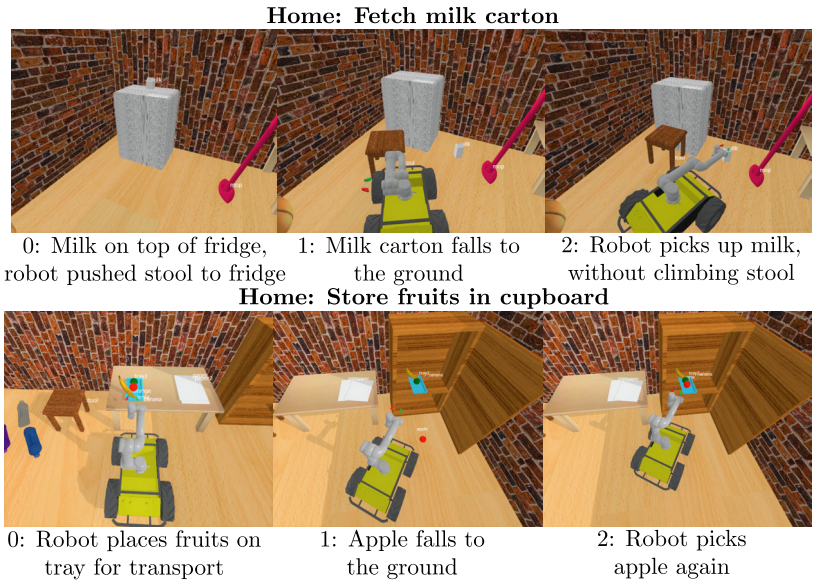


Figure 8: Interleaved action prediction and execution enables adaptation in case of unexpected errors during action execution. (top) robot recovers after the milk carton falls. (bottom) recovers after a fruit falls from the *tray* by picking it up again.

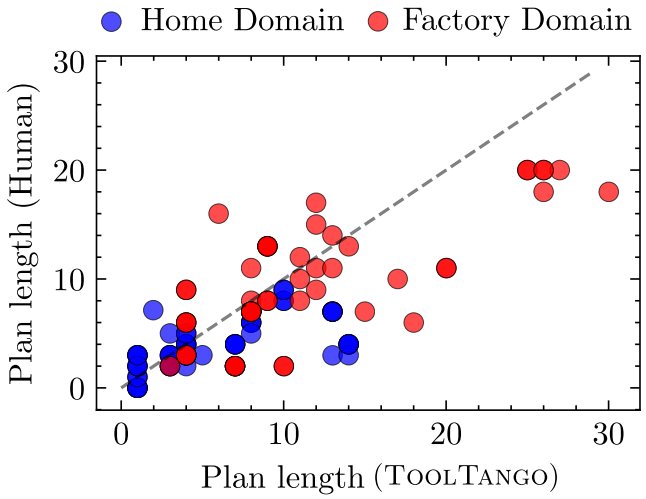


Figure 9: Scatter plot comparing the lengths of plans obtained from model predictions and those from human demonstrations. The plans generated from TOOLTANGO are of similar length as that of human demonstrators.

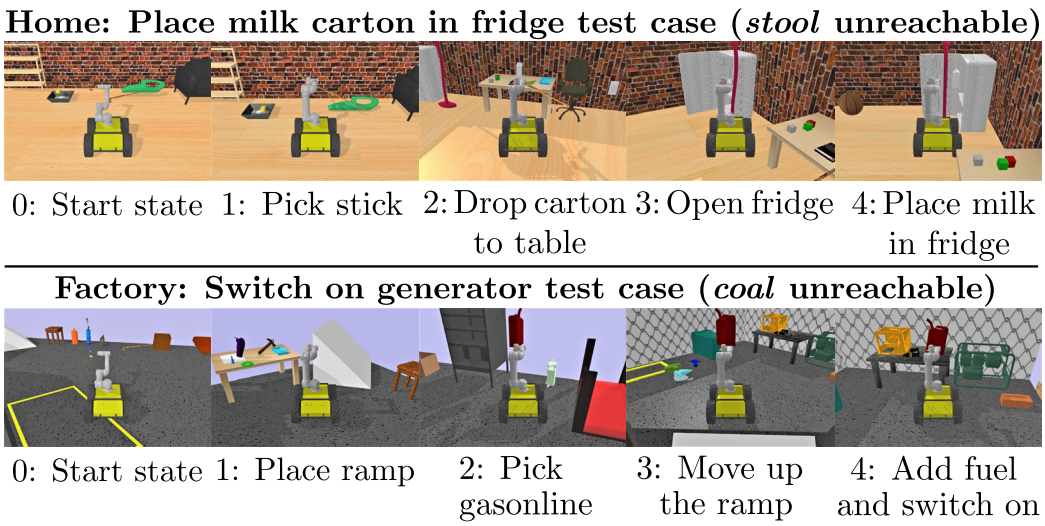


Figure 10: Test cases where TANGO fails but TOOLTANGO reaches the goal state. Typically in cases with complex and long plans with multiple tools being used. TANGO predicts *stool* in the first case and *coal* in the second case, both of which are unreachable. TOOLTANGO predicts reachable tools. (Top) Here the robot needs to place the milk carton which is at a higher location which is unreachable without a stool. In this test case, the stool in unreachable so the model uses a stick to reach the milk carton. (Bottom) Switching on the generator requires a fuel. In this test case coal is unreachable/unavailable and so the model instead uses gasoline to turn on the generator.

for the task of "storing away the fruits in the cupboard", the robot predicts the use of tray for the transport task. During execution the apple falls off the tray. The robot correctly re-collects the apple.

6.3.3 PLAN EFFICIENCY COMPARISON

Figure 9 compares the length of robot plans predicted by the learned model against human demonstrated plans. We observe that, on average, the predicted plan lengths are close to the human demonstrated ones. In 12% cases, the plans predicted by TOOLTANGO utilize tools satisfying the goal condition in fewer steps compared to the human demonstrated plan.

6.3.4 ANALYSIS OF INDEPENDENT TOOL PREDICTION

Figure 10 demonstrates the advantage of independent tool prediction and how it augments the TOOLTANGO to improve goal-reaching performance. The figure shows two cases where the TANGO model is unable to reach the goal state, but TOOLTANGO reaches a goal. The first example shows a setting where a milk carton is placed on top of the fridge and stool is kept directly in front of the fridge. In this case, TANGO predicts the use of *stool* to reach the carton; however, it first opens the fridge door, making *stool* inaccessible. On the other hand, TOOLTANGO predicts using a *stick* to drop the carton on top of the table and is able to place it inside the fridge. The second example shows a case where the generator needs to be powered on after adding a fuel source. Here, TOOLTANGO predicts the picking *coal* that is on top of the shelf. However, it does not use any tool to elevate itself and the execution returns an error of unreachable object. Instead, TOOLTANGO predicts the use of *gasoline* that is placed on the ground and is successfully able to execute the plan.

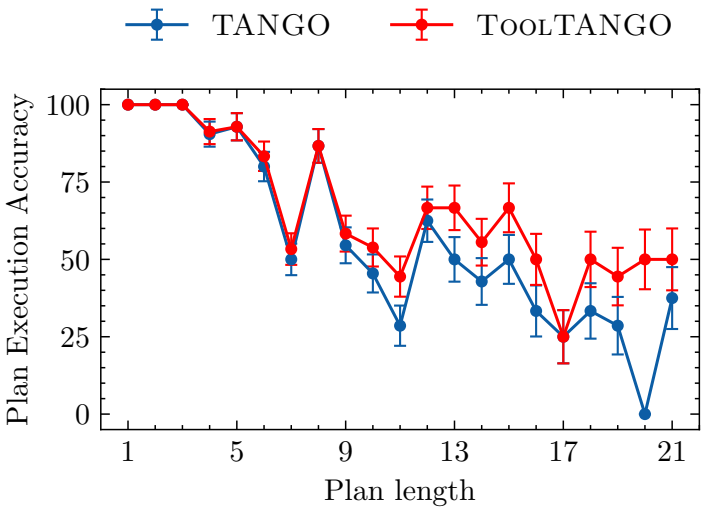


Figure 11: Execution accuracy of inferred plans with plan length for TANGO and TOOLTANGO. The latter gives a higher goal-reaching performance than the former for plans with longer action sequences.

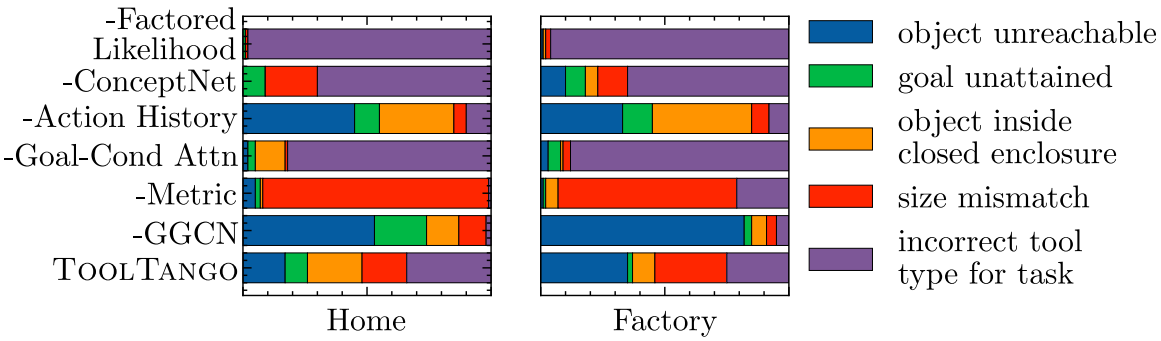


Figure 12: An analysis of fractional errors during plan execution using the learned TOOLTANGO model. The horizontal axis denotes the total errors for the ablated model. The absolute value of the total errors are shown in Table 4

Independent tool prediction also allows the model to perform better in complex settings requiring longer action sequences and using multiple tools. Figure 11 assesses the model accuracy with the lengths of the inferred plans. We observe that TOOLTANGO has a higher plan execution accuracy in case of (goal, scene) pairs where the average plan length to attain a goal state is high. For instance, if the goal is to place fruits in side the cupboard and all the fruits are on top of the fridge, the robot needs to use a tool to first reach the fruits and then use another tool to carry them to the cupboard. In this case the TANGO model directly tries to pick unreachable fruits or predicts an incorrect tool to reach the fruits. TOOLTANGO is able to execute complex task by subsequently using a *stool* and *tray*.

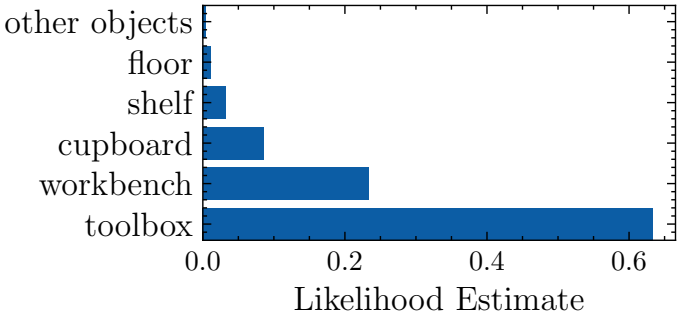


Figure 13: Likelihood estimates of exploration for different objects where "screw" might be found.

7. Limitations and Future Work

7.1 Scaling to Longer Plans

As we observed in Figure 11, the plan execution accuracy decreases by 20% on the Test sets and 30% on Generalization Test sets. This merits investigation into planning abstractions (Vega-Brown & Roy, 2020) for scaling to longer plan lengths in realistic domains. Figure 12 analyzes the errors encountered during plan execution using actions predicted by the proposed model. In 27% of the cases, the model misses a pre-requisite actions required for a pre-condition for initiating the subsequent action. For example, missing the need to open the door before exiting the room (object unreachable 19%) or missing opening the cupboard before picking an object inside it (object inside enclosure 8%). There is scope for improvement here by incorporating explicit causal structure (Nair, Zhu, Savarese, & Fei-Fei, 2019b).

7.2 Partially Observable Environments

In realistic scenarios, the robot's environment may only be partially-known due to the limited view and scope of the robot's sensors. For example, tools such as screws may be stored away in a tool box and may not be easily observed by the robot. In such a scenario, we expect the robot to learn to predict possible locations for exploration based on common sense knowledge. For example, the robot should explore the tool box. Further, the subgeometry information may be fed into the model to determine the set of objects in the environment. In order to address partially-observed worlds, we can extend the prediction model as follows. Instead of learning attention only over candidate objects, we can learn a *generalized* attention over spatial relations modeled in the graph network. Such an extension allows the model to predict a preference order for locations that the robot should explore to find the required tool. Figure 13 illustrates an example, where the robot shows an example, where the robot predicts that the screws may be found in toolbox or workbench. Please note that this result is indicative of the possibility of using the model in partially-known world and will be investigated in detail as part of future work.

7.3 Extension to Realistic Robot Control Settings

The action representations adopted in this work are inspired from task and motion planning communities (Zhu, Tremblay, Birchfield, & Zhu, 2021; Migimatsu & Bohg, 2020). We simulate a mobile manipulator (Clearpath Robotics Husky with a mounted Universal Robotics UR5 arm). The robot's

action space is modeled as high-level skills (or behaviors), each in-turn realized using a low-level motion plan or controller that is parameterized at run time. For navigation actions, a standard state-space planner (A* with Manhattan distance as a heuristic) for a coarse collision free path and realize point to point motion through a velocity-based controller. For the manipulation, we consider a 6 degree of freedom model of the manipulator. For object manipulation a crane-grasp was implemented where the arm would move to a pre-grasp position to the intended object of interest. The arm motion was realized using a joint state controller moving the arm to the end-effector pose. The precise manipulation of tools is abstracted as follows: once the robot end-effector makes contact a rigid joint is established for subsequent motions till it is released. As mentioned in Section 1, learning the precise manipulation of tools is not the focus of this work and can be delegated to a method such as the one proposed by Park et al. (2019). All robot actions are executed in the PyBullet physics engine with a mesh-based model of objects and a URDF-model for the robot manipulation system. The technical details appear in Appendix D. The entire implementation of the robot behaviors and the simulation environment is available at `https://github.com/reail-iitd/tango` for the use of the research community.

8. Conclusions

This paper proposes TOOLTANGO, a novel neural architecture that learns a policy to attain intended goals as tool interaction sequences leveraging fusion of semantic and metric representations, goal-conditioned attention, knowledge-base corpora. TOOLTANGO is trained using a data set of human instructed robot plans with simulated world states in home and factory like environments. It uses an independent model that predicts likelihood scores for each tool at each time-step. The imitation learner demonstrates accurate common sense generalization to environments with novel object instances using the learned knowledge of shared spatial and semantic characteristics. It also shows the ability to adapt to erroneous situations and stochasticity in action execution. Finally, TOOLTANGO synthesizes a sequence of tool interactions with a high accuracy of goal-attainment.

Acknowledgments

Most work done when the first two authors were undergraduate students at Indian Institute of Technology Delhi, India. Mausam is supported by an IBM SUR award, grants by Google, Bloomberg and 1MG, Jai Gupta chair fellowship, and a Visvesvaraya faculty award by Govt. of India. Rohan Paul acknowledges support from Pankaj Gupta Faculty Fellowship and DST's Technology Innovation Hub (TIH) for Cobotics. Shreshth Tuli is supported by the President's PhD Scholarship at Imperial College London. We thank the IIT Delhi HPC facility and Prof. Prem Kalra and Mr. Anil Sharma at the CSE VR Lab for compute resources. We thank Mr. Pulkit Sapra and Prof. P. V. M. Rao for assistance with CAD model creation. We thank Puig et al. (2018) for sharing the implementation for baseline comparison. We are grateful to anonymous turkers and student volunteers for assisting in the data collection.

Appendix A. Supplementary Material

All our code is available as a GitHub repository under BSD-2 License `https://github.com/reail-iitd/tango`. Instructions for reproducing the results are given at `https://github.`

com/reail-iitd/tango/wiki. A supplementary video demonstrating our data collection platform and model's generalization capability is available at https://www.youtube.com/watch?v=lUWU3rK1Gno.

Appendix B. Hyperparameter Details

We detail the hyper-parameters for the TOOLTANGO architecture introduced in this paper.

- *Graph Structured World Representation.* The Gated Graph Convolution Network (GGCN) was implemented with 4-hidden layers, each of size 128, with convolutions across 2 time steps for every relation passing through a layer normalized GRU cell. The Parameterized ReLU activation function with a 0.25 negative input slope was used in all hidden layers.

- *Word Embeddings.* The word embeddings (derived from $\mathrm{ConceptNet}$) were of size 300. Additionally, the semantic state of each object was encoded as a one-hot vector of size 29. Typically, there were 35 and 45 objects in the home and factory domains respectively.

- *Fusing Metric Information.* The metric encodings were generated from the metric information associated with objects using a 2-layer Fully Connected Network (FCN) with 128-sized layers.

- *Encoding Action History.* A Long Short Term Memory (LSTM) layer of size 128 was used to encode the action history using the generalized action encoding $\mathcal{A}(I_t(o_t^1, o_t^2))$.

- *Goal-conditioned Attention.* The attention network was realized as a 1-layer FCN of layer size 128 with a $\mathrm{softmax}$ layer at the end.

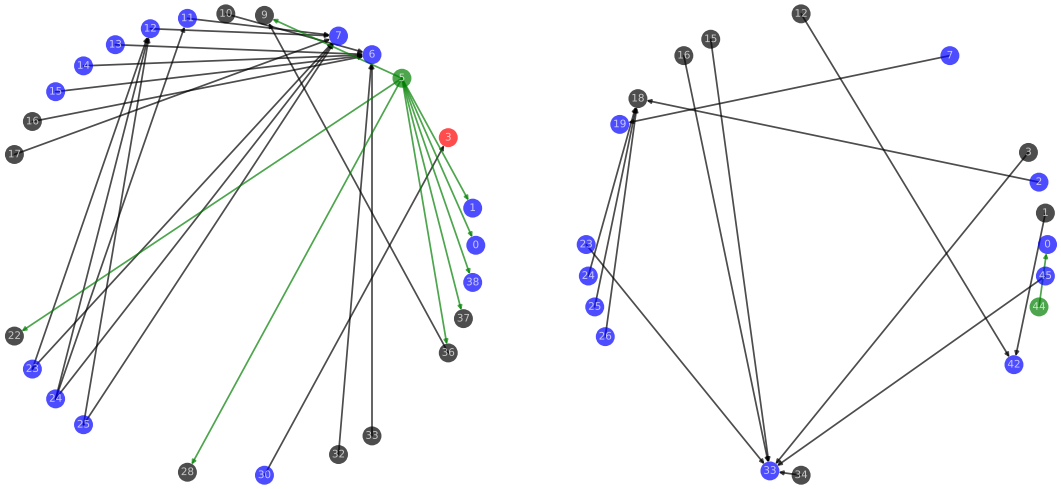
- *Tool Prediction.* To predict the tool-likelihood score p_t, a 3-layer FCN was used, each hidden layer with size 128 and output layer with size 1 with sigmoid activation function.

- *Action Prediction.* To predict the action I_t, a 3-layer FCN was used, each hidden layer with size 128 and output layer with size $|\mathcal{I}|$. I_t was converted to a one-hot encoding $\vec{I_t}$. This, with the object embedding e_o was passed to the o_t^1 predictor via an FCN. This FCN consists of 3-hidden layers of size 128 and a final layer of size 1 with a sigmoid activation (for likelihood). The $\vec{I_t}$ and o_t^1 likelihoods were sent to the o_t^2 predictor to predict likelihoods for all object embeddings e_o. This part was realized as a 3-layer FCN with hidden layer size 128 and final layer of size 1 with a sigmoid activation function.

- *Training parameters.* Model training used a learning rate of 5×10^{-4}. The Adam optimizer (Kingma & Ba, 2014) with a weight decay parameter of 10^{-5} and a batch size of 1 was used. An early stopping criterion was applied for convergence. The *action prediction accuracy* was used as the comparison metric on the validation set or up to a maximum of 200 epochs.

Appendix C. World Scenes

The figure 14 illustrates the object-centric graph representation of the 10 world scenes we used for Home and Factory domains. The *agent node* is shown in green, *tools* in black and *objects with states*



(a) A sample home scene. (b) A sample factory scene.

Figure 14: **Sample World Scenes in Home and Factory domains**

in red. The relations of *Close* are shown in green (only populated for agent), *On* in black, *Inside* in red and *Stuck to* in blue.

The legend for node IDs to objects are given below:

Home: 0: floor, 1: walls, 2: door, 3: fridge, 4: cupboard, 5: husky, 6: table, 7: table2, 8: couch, 9: big-tray, 10: book, 11: paper, 12: paper2, 13: cube gray, 14: cube green, 15: cube red, 16: tray, 17: tray2, 18: light, 19: bottle blue, 20: bottle gray, 21: bottle red, 22: box, 23: apple, 24: orange, 25: banana, 26: chair, 27: ball, 28: stick, 29: dumpster, 30: milk, 31: shelf, 32: glue, 33: tape, 34: stool, 35: mop, 36: sponge, 37: vacuum, 38: dirt.

Factory: 0: floor warehouse, 1: 3D printer, 2: assembly station, 3: blow dryer, 4: board, 5: box, 6: brick, 7: coal, 8: crate green, 9: crate peach, 10: crate red, 11: cupboard, 12: drill, 13: gasoline, 14: generator, 15: glue, 16: hammer, 17: ladder, 18: lift, 19: long shelf, 20: mop, 21: nail, 22: oil, 23: paper, 24: part1, 25: part2, 26: part3, 27: platform, 28: screw, 29: screwdriver, 30: spraypaint, 31: stick, 32: stool, 33: table, 34: tape, 35: toolbox, 36: trolley, 37: wall warehouse, 38: water, 39: welder, 40: wood, 41: wood cutter, 42: worktable, 43: ramp, 44: husky, 45: tray.

Appendix D. Realization of Object Relations

To implement various relations among objects PyBullet allows to define *manipulation regions* around objects. These regions are virtual regions inside which a robot can manipulate that object. A visual description of this region is shown in Figure 15.

We now define two object types.

1. **Standalone object:** Such an object is defined as a collection of stereo-lithographic tetrahedrons with visual and collision properties. Each such object has a single manipulation region, denoted as $MR(object)$, which is the region around the object where it is considered to be in

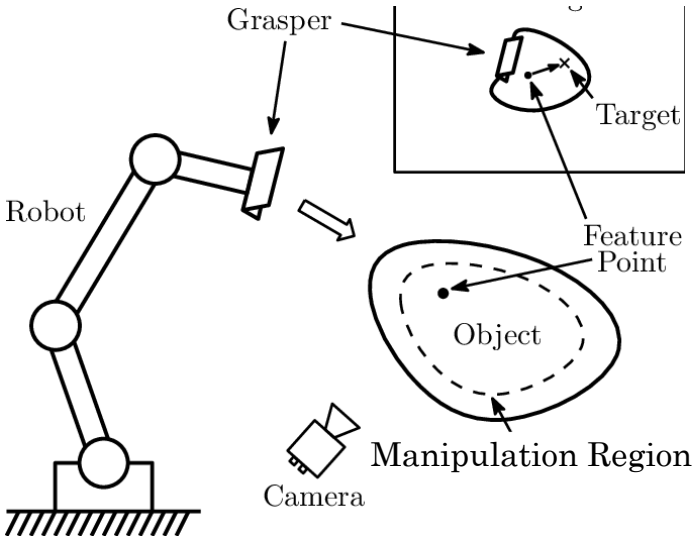


Figure 15: Manipulation region of an object

"vicinity" to the object and hence can be manipulated by the robot. Examples: cubes, fruits, milk, etc.

2. **Containing object:** Such an object is a standalone object with an additional containment region, denoted as $CR(object)$, which is a proper subset of the manipulation region where if another object's center lies is called to be "contained inside" this object. Examples: box, cupboard and fridge.

Every object has a center defined arithmetic mean of the centers of the tetrahedrons, denoted by $C(object)$. We now define the two relation constraints of which others are different versions of combinations of:

1. **Inside:** An object A is said to be contained inside a containing object B if the $C(A) \in CR(B)$.
2. **On Top:** An object A is said to be on top on another object B if $C(A)_z > C(B)_z$ and $MR(A) \cap MR(B) \neq \phi$, where $C(O)_z$ denotes the z component of the center of object O.

References

Allen, K. R., Smith, K. A., & Tenenbaum, J. (2019). Rapid trial-and-error learning in physical problem solving.. In *CogSci*, p. 90.

Antunes, A., Saponaro, G., Dehban, A., Jamone, L., Ventura, R., Bernardino, A., & Santos-Victor, J. (2015). Robotic tool use and problem solving based on probabilistic planning and learned affordances. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. Workshop Learn. Object Affordances Fundamental Step to Allow Prediction Plan. Tool use*.

Bae, H., Kim, G., Kim, J., Qian, D., & Lee, S. (2019). Multi-robot path planning method using reinforcement learning. *Applied Sciences*, *9*(15), 3057.

Bahdanau, D., Cho, K. H., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR*

2015.

Bansal, R., Tuli, S., Paul, R., & Mausam (2020). Toolnet: Using commonsense generalization for predicting tool use for robot plan synthesis. In *Workshop on Advances & Challenges in Imitation Learning for Robotics at Robotics Science and Systems (RSS)*.

Bisk, Y., Zellers, R., Bras, R. L., Gao, J., & Choi, Y. (2020). Piqa: Reasoning about physical commonsense in natural language. In *AAAI*.

Boteanu, A., Kent, D., Mohseni-Kabir, A., Rich, C., & Chernova, S. (2015). Towards robot adaptability in new situations. In *2015 AAAI Fall Symposium Series*. Citeseer.

Calli, B., Singh, A., Bruce, J., Walsman, A., Konolige, K., Srinavasa, S. S., Abbeel, P., & Dollar, A. M. (2017). YCB Benchmarking Project: Object Set, Data Set and Their Applications. *Journal of The Society of Instrument and Control Engineers*, *56*(10), 792–797.

Chen, H., Tan, H., Kuntz, A., Bansal, M., & Alterovitz, R. (2020). Enabling robots to understand incomplete natural language instructions using commonsense reasoning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1963–1969. IEEE.

Choi, D., Langley, P., & To, S. T. (2018). Creating and using tools in a hybrid cognitive architecture.. In *AAAI Spring Symposia*.

Coumans, E., & Bai, Y. (2016). Pybullet, a python module for physics simulation for games, robotics and machine learning. In *GitHub repository*.

Driess, D., Oguz, O., Ha, J.-S., & Toussaint, M. (2020). Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9563–9569. IEEE.

Finn, C., Yu, T., Zhang, T., Abbeel, P., & Levine, S. (2017). One-shot visual imitation learning via meta-learning. In *Conference on Robot Learning*, pp. 357–368. PMLR.

Fitzgerald, T., Goel, A., & Thomaz, A. (2018). Human-guided object mapping for task transfer. *ACM Transactions on Human-Robot Interaction (THRI)*, *7*(2), 1–24.

Fitzgerald, T., Goel, A., & Thomaz, A. (2021). Modeling and learning constraints for creative tool use. *Frontiers in Robotics and AI*, *8*.

Gajewski, P., et al. (2019). Adapting everyday manipulation skills to varied scenarios. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 1345–1351. IEEE.

Garg, S., Bajpai, A., & Mausam (2019). Size independent neural transfer for RDDL planning. In Benton, J., Lipovetzky, N., Onaindia, E., Smith, D. E., & Srivastava, S. (Eds.), *Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling, ICAPS 2018, Berkeley, CA, USA, July 11-15, 2019*, pp. 631–636. AAAI Press.

Garg, S., Bajpai, A., & Mausam (2020). Symbolic network: Generalized neural policies for relational mdps. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 3397–3407. PMLR.

Garrett, C. R., et al. (2021). Integrated task and motion planning. *Annu. Rev. Control Robot. Auton. Syst*, *2021*(4), 1–30.

Garrett, C. R., Lozano-Pérez, T., & Kaelbling, L. P. (2020). Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, Vol. 30, pp. 440–448.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034.

Hermans, T., Rehg, J. M., & Bobick, A. (2011). Affordance prediction via learned object attributes. In *ICRA: Workshop on Semantic Perception, Mapping, and Exploration*, pp. 181–184.

Holladay, R., Lozano-Pérez, T., & Rodriguez, A. (2019). Force-and-motion constrained planning for tool use. In *IROS*.

Huang, D.-A., Nair, S., Xu, D., Zhu, Y., Garg, A., Fei-Fei, L., Savarese, S., & Niebles, J. C. (2019). Neural task graphs: Generalizing to unseen tasks from a single video demonstration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8565–8574.

Huang, S. H., Pan, J., Mulcaire, G., & Abbeel, P. (2015). Leveraging appearance priors in non-rigid registration, with application to manipulation of deformable objects. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 878–885. IEEE.

Hübner, J. F., Bordini, R. H., & Wooldridge, M. (2006). Programming declarative goals using plan patterns. In *International Workshop on Declarative Agent Languages and Technologies*, pp. 123–140. Springer.

Jain, A., Das, D., Gupta, J. K., & Saxena, A. (2015). Planit: A crowdsourcing approach for learning to plan paths from large scale preference feedback. In *ICRA*, pp. 877–884.

Kho, G., Hung, C., & Cunningham, H. (2014). Robo brain: Massive knowledge base for robots. In *Cornell Univ., USA, Tech. Rep*.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. In *CoRR arXiv:1412.6980*.

Kolobov, A., Mausam, Weld, D. S., & Geffner, H. (2011). Heuristic search for generalized stochastic shortest path mdps. In *ICAPS*.

Kroemer, O., Ugur, E., Oztop, E., & Peters, J. (2012). A kernel-based approach to direct action perception. In *2012 IEEE international Conference on Robotics and Automation*, pp. 2605–2610. IEEE.

Lee, A. X., Gupta, A., Lu, H., Levine, S., & Abbeel, P. (2015). Learning from multiple demonstrations using trajectory-aware non-rigid registration with applications to deformable object manipulation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5265–5272. IEEE.

Levihn, M., & Stilman, M. (2014). Using environment objects as tools: Unconventional door opening. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2502–2508. IEEE.

Li, Y., Tarlow, D., Brockschmidt, M., & Zemel, R. (2015). Gated graph sequence neural networks. In *arXiv preprint arXiv:1511.05493*.

Liao, Y.-H., Puig, X., Boben, M., Torralba, A., & Fidler, S. (2019). Synthesizing environment-aware activities via activity sketches. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6291–6299.

Liu, Z., Freeman, W. T., Tenenbaum, J. B., & Wu, J. (2018). Physical primitive decomposition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–19.

Lynch, C., Khansari, M., Xiao, T., Kumar, V., Tompson, J., Levine, S., & Sermanet, P. (2020). Learning latent plans from play. In *Conference on Robot Learning*, pp. 1113–1132. PMLR.

Mandlekar, A., Zhu, Y., Garg, A., Booher, J., Spero, M., Tung, A., Gao, J., Emmons, J., Gupta, A., Orbay, E., et al. (2018). Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pp. 879–893. PMLR.

Mausam, & Kolobov, A. (2012). Planning with Markov decision processes: An AI perspective. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *6*(1), 1–210.

Migimatsu, T., & Bohg, J. (2020). Object-centric task and motion planning in dynamic environments. *IEEE Robotics and Automation Letters*, *5*(2), 844–851.

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2018). Advances in pre-training distributed word representations. In *LREC*.

Misra, D. K., Sung, J., Lee, K., & Saxena, A. (2016). Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. *IJRR*, *35*(1-3), 281–300.

Myers, A., Teo, C. L., Fermüller, C., & Aloimonos, Y. (2015). Affordance detection of tool parts from geometric features. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1374–1381. IEEE.

Nair, A., Chen, D., Agrawal, P., Isola, P., Abbeel, P., Malik, J., & Levine, S. (2017). Combining self-supervised learning and imitation for vision-based rope manipulation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2146–2153. IEEE.

Nair, L., & Chernova, S. (2020). Feature guided search for creative problem solving through tool construction. In *Frontiers in Robotics and AI*, p. 205. Frontiers.

Nair, L., Srikanth, N. S., Erickson, Z. M., & Chernova, S. (2019a). Autonomous tool construction using part shape and attachment prediction.. In *Robotics: Science and Systems*.

Nair, S., Zhu, Y., Savarese, S., & Fei-Fei, L. (2019b). Causal induction from visual observations for goal directed tasks. In *CoRR arXiv:1910.01751*.

Nyga, D., & Beetz, M. (2018). Cloud-based probabilistic knowledge services for instruction interpretation. In *Robotics Research*, pp. 649–664. Springer.

Nyga, D., Roy, S., Paul, R., Park, D., Pomarlan, M., Beetz, M., & Roy, N. (2018). Grounding robot plans from natural language instructions with incomplete world knowledge. In *Conference on Robot Learning*, pp. 714–723.

Park, D., Noseworthy, M., Paul, R., Roy, S., & Roy, N. (2019). Inferring task goals and constraints using bayesian nonparametric inverse reinforcement learning. In *Proceedings of the 3rd Conference on Robot Learning (CoRL)*.

Puig, X., Ra, K., Boben, M., Li, J., Wang, T., Fidler, S., & Torralba, A. (2018). Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8494–8502.

Sarathy, V., & Scheutz, M. (2018). Macgyver problems: Ai challenges for testing resourcefulness and creativity. *Advances in Cognitive Systems*, *6*, 31–44.

Scalise, R., Li, S., Admoni, H., Rosenthal, S., & Srinivasa, S. S. (2018). Natural language instructions for human–robot collaborative manipulation. *The International Journal of Robotics Research*, *37*(6), 558–565.

Sharma, S., Gupta, J., Tuli, S., Paul, R., & Mausam (2022a). Goalnet: Inferring conjunctive goal predicates from human plan demonstrations for robot instruction following. In *International Conference on Automated Planning and Scheduling (ICAPS) - Planning and Reinforcement Learning Workshop*.

Sharma, V., Arora, D., Geißer, F., Mausam, & Singla, P. (2022b). Symnet 2.0: Effectively handling non-fluents and actions in generalized neural policies for RDDL relational MDPs. In *Proceedings of the 38th Conference on Uncertainty in Artificial Intelligence (UAI)*.

Shridhar, M., Thomason, J., Gordon, D., Bisk, Y., Han, W., Mottaghi, R., Zettlemoyer, L., & Fox, D. (2020). Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10740–10749.

Silver, T., et al. (2021). Planning with learned object importance in large problem instances using graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, pp. 11962–11971.

Speer, R., Chin, J., & Havasi, C. (2017). ConceptNet 5.5: An open multilingual graph of general knowledge. In *AAAI*.

Speer, R., Chin, J., & Havasi, C. (2019). ConceptNet Numberbatch, the best pre-computed word embeddings you can use. In *GitHub repository*.

Srivastava, S., Fang, E., Riano, L., Chitnis, R., Russell, S., & Abbeel, P. (2014). Combined task and motion planning through an extensible planner-independent interface layer. In *2014 IEEE international conference on robotics and automation (ICRA)*, pp. 639–646. IEEE.

Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J. J., Mur-Artal, R., Ren, C., Verma, S., et al. (2019). The replica dataset: A digital replica of indoor spaces. In *CoRR arXiv:1906.05797*.

Toussaint, M. A., Allen, K. R., Smith, K. A., & Tenenbaum, J. B. (2018). Differentiable physics and stable modes for tool-use and manipulation planning. In *Robotics: Science and Systems Foundation*.

Tuli, S., Bansal, R., Paul, R., & Mausam (2021). Tango: Commonsense generalization in predicting tool interactions for mobile manipulators. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Vega-Brown, W., & Roy, N. (2020). Asymptotically optimal planning under piecewise-analytic constraints. In *Algorithmic Foundations of Robotics XII*, pp. 528–543. Springer.

Wu, J., Lim, J. J., Zhang, H., Tenenbaum, J. B., & Freeman, W. T. (2016). Physics 101: Learning physical object properties from unlabeled videos. In *British Machine Vision Conference*.

Wu, J., Yildirim, I., Lim, J. J., Freeman, B., & Tenenbaum, J. (2015). Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In Cortes, C., Lawrence,

N. D., Lee, D. D., Sugiyama, M., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems 28*, pp. 127–135. Curran Associates, Inc.

Xie, A., Ebert, F., Levine, S., & Finn, C. (2019). Improvisation through physical understanding: Using novel objects as tools with visual foresight. In *Robotics at Robotics Science and Systems (RSS)*.

Zhu, Y., Tremblay, J., Birchfield, S., & Zhu, Y. (2021). Hierarchical planning for long-horizon manipulation with geometric and symbolic scene graphs. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6541–6548. IEEE.