GraspGPT: Leveraging Semantic Knowledge From a Large Language Model for Task-Oriented Grasping

Chao Tang, Dehao Huang, Wenqi Ge, Weiyu Liu, and Hong Zhang

***Abstract*—Task-oriented grasping (TOG) refers to the problem of predicting grasps on an object that enable subsequent manipulation tasks. To model the complex relationships between objects, tasks, and grasps, existing methods incorporate semantic knowledge as priors into TOG pipelines. However, the existing semantic knowledge is typically constructed based on closed-world concept sets, restraining the generalization to novel concepts out of the pre-defined sets. To address this issue, we propose GraspGPT, a large language model (LLM) based TOG framework that leverages the open-end semantic knowledge from an LLM to achieve zero-shot generalization to novel concepts. We conduct experiments on Language Augmented TaskGrasp (LA-TaskGrasp) dataset and demonstrate that GraspGPT outperforms existing TOG methods on different held-out settings when generalizing to novel concepts out of the training set. The effectiveness of GraspGPT is further validated in real-robot experiments.**

***Index Terms*—Grasping, perception for grasping and manipulation, deep learning in grasping and manipulation.**

I. Introduction

TOOL manipulation is a fundamental skill for household robots. To achieve successful tool manipulation and accomplish specific goals, the robot must, in the first place, grasp the tool in a task-oriented manner, i.e., perform task-oriented grasping [1], [2]. For instance, accurately gripping the handle of a knife to slice an apple into pieces or securely holding the tip of the blade of the knife during a handover. Considering the vast number of object classes and tasks in open-world operating environments like offices and kitchens, it is challenging to model the complex relationships between object classes, tasks, and grasps due to the diverse and dynamic nature of open-world environments. Practically, one can expect the robot to be trained on a limited set of examples and generalize the learned TOG skills to novel object classes and tasks beyond the training examples.

To achieve such a goal, recent works have proposed incorporating semantic knowledge into TOG pipelines to enable robots to adapt to various situations. Semantic knowledge provides high-level abstractions of open-world environments and captures the underlying relationships between concepts. For instance, Song et al. [3] construct a semantic knowledge base (KB) with a pre-defined set of concepts and constraints with Bayesian Networks. Recently, Murali et al. [4] contribute the largest and the most diverse TOG dataset, named TaskGrasp dataset, and build a knowledge graph (KG) based on the concepts collected in the dataset. Although these methods have demonstrated their generalization abilities to concepts pre-defined within the KB, they still operate under the closed-world assumption and cannot handle novel concepts out of the KB. This limitation is critical as a household robot must deal with open-end object classes and tasks.

The recent advancements in large language models (LLMs) [5], [6] have brought about significant progresses in various robot tasks [7], [8], [9], [10]. These LLMs are trained with internet-scale text corpora. Thus, robots can seamlessly extract and harness open-end semantic knowledge from LLMs to plan actions in unseen scenarios. In this letter, we follow the same spirit and introduce **GraspGPT**, an LLM-based TOG framework. GraspGPT distinguishes itself from previous TOG methods by not being constrained to a closed-world concept set. Instead, it leverages the open-end semantic knowledge about object classes and tasks from an LLM to achieve zero-shot generalization to novel concepts out of the training set. Specifically, we focus on two types of concepts: object class and task. As is shown in Fig. 1(a), when presented with a novel concept in a language instruction, GraspGPT first prompts an LLM to acquire a set of natural language description paragraphs of the concept. These description paragraphs connect the novel concept to its related concepts described during training, as depicted in Fig. 1(b). Subsequently, the robot can generalize the learned TOG skills from known concepts to novel concepts out of the training set. Evaluation on the contributed TOG dataset named Language Augmented TaskGrasp (LA-TaskGrasp) dataset demonstrates that GraspGPT outperforms existing TOG methods under different held-out settings. We further deploy GraspGPT on a Kinova Gen3 robotic arm to validate its effectiveness in real-world robotic applications.

In summary, our contributions are two-fold:

- We propose GraspGPT, an LLM-based TOG framework that leverages the open-end semantic knowledge from an

Manuscript received 24 July 2023; accepted 18 September 2023. Date of publication 27 September 2023; date of current version 6 October 2023. This letter was recommended for publication by Associate Editor R. Martin-Martin and Editor H. Liu upon evaluation of the reviewers' comments. This work was supported by the Shenzhen Key Laboratory of Robotics and Computer Vision under Grant ZDSYS20220330160557001. *(Corresponding author: Chao Tang.)*

Chao Tang, Dehao Huang, Wenqi Ge, and Hong Zhang are with the Shenzhen Key Laboratory of Robotics and Computer Vision, Southern University of Science and Technology, Shenzhen 518055, China, and also with the Department of Electronic and Electrical Engineering, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: 12131042@mail.sustech.edu.cn; huangdehao919@gmail.com; 12232112@mail.sustech.edu.cn; hzhang@sustech.edu.cn).

Weiyu Liu is with the Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: loftusoklamtin69@gmail.com).

Our code, data, appendix, and video are publicly available at https://sites.google.com/view/graspgpt.

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3320012, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3320012

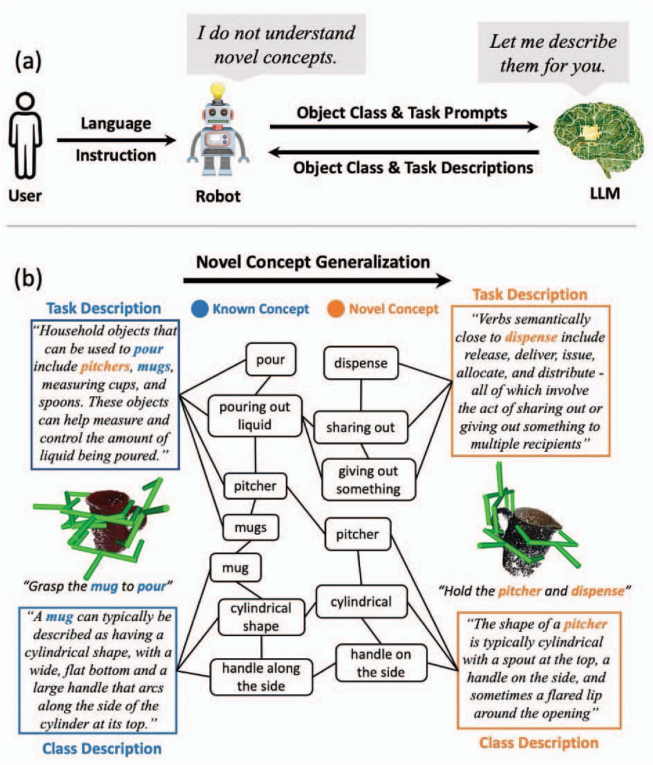

Fig. 1. (a) GraspGPT prompts an LLM to acquire language descriptions about the novel concept(s) in a natural language instruction given by the user. (b) Language descriptions connect the novel concept to its related concepts described during training, enabling the generalization of task-oriented grasping skills from known concepts to novel concepts.

LLM to achieve zero-shot generalization to novel concepts out of the training set.

- We present a pipeline to automatically generate language descriptions of concepts with an LLM and contribute a language augmented TOG dataset named LA-TaskGrasp dataset.

II. Related Work

A. Task-Oriented Grasping

The ability to perform task-oriented grasping is essential for household robots as it is the first step towards tool manipulation. Data-driven approaches have achieved success in solving TOG problems to some extent. Dang et al. [11] and Liu et al. [12] propose novel semantic representations of grasp contexts for task-oriented grasp pose prediction. These methods learn class-task-grasp relationships purely from data without any external knowledge sources, thus achieving unsatisfying performance. More recent works [3], [4], [13], [14] have proposed incorporating semantic knowledge as priors into TOG pipelines. Song et al. [3] construct a semantic KB with a set of tasks, object classes, actions, constraints, and reason over the KB using Bayesian Networks. Similarly, Ardón et al. [13] and Antanas et al. [14] build KGs relating pre-defined semantic attributes using probabilistic logic approaches. Despite these advancements, a significant bottleneck that hinders the generalization to a broader range of object classes and tasks is the need for large-scale TOG datasets. Motivated by this dilemma, Murali et al. [4] contribute the largest and the most diverse TOG dataset, named TaskGrasp dataset, and build a KG based on the concepts collected in the dataset. More importantly, they propose the state-of-the-art TOG algorithm GCNGrasp, which builds upon the semantic knowledge encoded in the KG, to generalize to concepts within the KG. However, the major limitation of GCNGrasp is its inability to directly handle novel concepts out of the graph. This limitation is critical as a household robot must deal with open-end object classes and tasks in real-world applications. In this letter, we address this problem by leveraging the open-end semantic knowledge from an LLM to generalize learned TOG skills to novel concepts.

B. LLMs in Robotics

Recent advances in LLMs have motivated the robotics community to harness the semantic knowledge embedded in these models for a wide range of robotic applications, such as tabletop manipulation [10], navigation [15], [16], and mobile manipulation [7], [17].

Huang et al. [9] first propose to decompose high-level tasks into mid-level plans with LLMs for robot decision making. To enable LLM-based robots to act properly in real-world applications, Ahn et al. [7] ground LLMs through affordance functions of pre-trained skills. Meanwhile, Huang et al. [18] extend previous work to include closed-loop feedback for both mobile and tabletop manipulation with a collection of perception models. Liang et al. [10] re-purpose LLMs to directly generate policy code running on real-world robots. More recent works [19], [20] combine multi-modal reasoning with LLMs for object rearrangement tasks. While aforementioned methods primarily consider LLMs for high-level task and motion planning, GraspGPT directly grounds the semantic knowledge from an LLM to grasping actions, which opens up the potential for optimizing other low-level policies (e.g., manipulation, navigation) with an LLM.

III. Problem Formulation

We assume access to an object class set $\mathcal{C} = \{c_i\}_{i=1}^{K_c}$ and a task set $\mathcal{T} = \{t_j\}_{j=1}^{K_t}$, where K_c and K_t are numbers of object classes and tasks, respectively. Based on $\mathcal{C}$ and $\mathcal{T}$, we consider the problem of learning task-oriented grasp pose prediction for a parallel-jaw gripper given the partial point cloud of an object $X_o \in \mathbb{R}^{N\times 3}$ and a natural language instruction I specifying an object class c and a task t, where N is the number of points. During training, we have $c = c_i \in \mathcal{C}$ and $t = t_j \in \mathcal{T}$. The challenge for real-world robotic applications is that c or t can be novel concepts (i.e., out of the training set) from open-world concept sets $\mathcal{C}_{ow}$ and $\mathcal{T}_{ow}$ during inference, where $\mathcal{C} \subset \mathcal{C}_{ow}$ and $\mathcal{T} \subset \mathcal{T}_{ow}$. To enable the generalization of TOG skills from known to novel concepts, GraspGPT incorporates open-end semantic knowledge by prompting an LLM to generate a set of object class description paragraphs L_c for c and a set of task description paragraphs L_t for t.

Mathematically, we aim to estimate the posterior distribution $P(G|X_o, I, L_c, L_t)$, where G represents the space of all task-oriented grasp poses. Following the convention in prior work [4], [21], the estimation process is factorized into two steps: 1) task-agnostic grasp sampling $P(G|X_o)$ and 2) task-oriented grasp evaluation $P(S|X_o, I, L_c, L_t, g)$, where S is the score (probability of success) for each $g \in G$. Each grasp pose g is represented by $(R, T) \in SE(3)$, where $R \in SO(3)$ represents

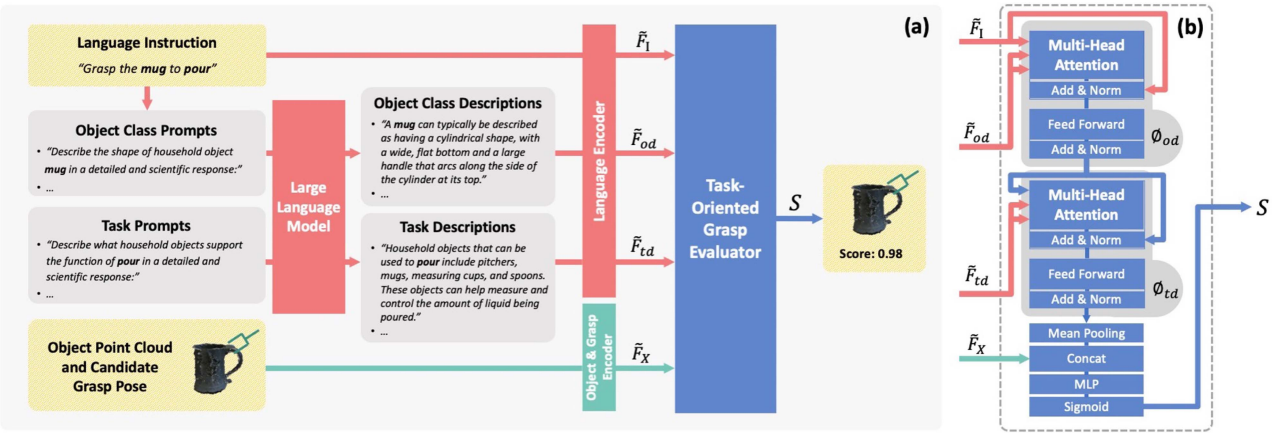

Fig. 2. (a) Overview of GraspGPT framework: when presented with a novel concept, such as a novel object class or task, in the natural language instruction, GraspGPT first prompts an LLM to acquire a set of language description paragraphs of the concept. Subsequently, GrasGPT evaluates the task compatibility of grasp candidates based on the multi-modal inputs from the sensors and an LLM. (b) The detailed structure of task-oriented grasp evaluator: the module is a customized transformer decoder that injects semantic knowledge from an LLM into the natural language instruction.

the 3D orientation and $T \in \mathbb{R}^3$ represents the 3D translation. Since the first step is well-studied by previous works, we directly apply off-the-shelf task-agnostic grasp sampler from [22] to obtain a set of grasp pose candidates, and focus on solving the second step.

IV. GraspGPT

A. Overview

An overview of the proposed GraspGPT framework is presented in Fig. 2(a). We begin by outlining the methodology for data generation in Section IV-B. We then detail the strategy for obtaining feature representations of multi-modal inputs in Section IV-C. Finally, we describe how to perform task-oriented grasp evaluation in Section IV-D.

B. Data Generation With an LLM

The data generation pipeline is designed based on TaskGrasp dataset that comprises K_c household object classes and K_t everyday tasks. To construct the Language Augmented TaskGrasp (LA-TaskGrasp) dataset, we employ an LLM to generate language descriptions for each object class c_i and task t_j, which will be described first. We then present the procedure for generating language instructions.

Language Description Generation The key idea behind GraspGPT is to leverage the language descriptions from an LLM to establish connections between novel and known concepts. According to Rosch's theory [23] of cognitive representations of semantic categories, a concept shares similar geometry, function, or effect descriptions with its related concepts. Inspired by [4], the similarity between concepts can be described by 1) directly prompting the LLM (e.g., *"Describe what verbs are similar to cut:"*) and 2) comparing the descriptions of two concepts (e.g., *"Describe the geometry of a cup/bowl:"*). In essence, the ability to relate concepts in this way is guaranteed by the fact that LLMs are trained on internet-scale data, enabling them to capture a broad range of linguistic patterns and semantic information.

To obtain L_{c_i} for object class c_i, we design two prompt sets: 1) property prompt set, each of which asks a property of c_i, and 2) similarity prompt set, each of which asks classes sharing a similar property with c_i. Here, properties can be shape, geometry, function, etc. Similarly, to obtain L_{t_j} for task t_j, we design: 1) affordance prompt set, each of which asks object classes that afford t_j, and 2) relevance prompt set, each of which asks semantically or physically relevant tasks to t_j. For each prompt set, we equally define N_p prompts. To generate language descriptions of concepts, we recursively query the LLM to generate N_a different answers per prompt. Examples of generated language descriptions are presented in Table I. We then orderly combine answers from each prompt to obtain complete description paragraphs of a concept, resulting in ${N_a}^{2N_p}$ description paragraphs. However, we have empirically observed that using only a subset of these paragraphs is sufficient for training due to the information redundancy. Each paragraph has an approximate length of 4-6 sentences. A complete list of prompts used to construct LA-TaskGrasp dataset and more examples of language descriptions can be found in the appendix. GraspGPT is not restricted to concepts defined in LA-TaskGrasp dataset, as it can incorporate open-end semantic knowledge from an LLM. It is a primary advantage over existing methods.

Language Instruction Generation To efficiently generate language instructions during each training loop, we employ a template-based generation strategy. Following our prior work [24], we begin with M templates from [25], such as *"Use the [obj] to [task]"*. Each template requires an object class label c_i and a task label t_j. To further enrich the vocabulary and grammatical diversities, we perform template augmentation using an LLM (e.g., *"rewrite the following sentence in a different grammatical format:"*) to generate M^+ additional templates, such as *"hold the [obj] in your hand and [task]"* and *"grip the [obj] in a [task]ing-friendly manner"*. For a complete list of $M + M^+$ templates used in LA-TaskGrasp dataset, please refer to the appendix. During each training loop, we randomly sample c_i and t_j from $\mathcal{C}$ and $\mathcal{T}$, and a template to generate a natural language instruction without human effort.

C. Multi-Modal Feature Representation

To incorporate semantic knowledge about concepts in the form of language descriptions into GraspGPT framework, we

TABLE I
EXAMPLES OF OBJECT CLASS AND TASK DESCRIPTIONS IN LA-TASKGRASP DATASET

Class	Property Description	Similarity Description
Mug	*"The mug is cylindrical in shape, with a slightly rounded base leading up to straight walls which eventually taper slightly towards the rim."*	*"Mugs typically have a cylindrical shape with a slightly tapered "top and a curved handle; objects with similar shapes include bottles and vases."*
Task	**Affordance Description**	**Relevance Description**
Sweep	*"Household objects that can be used to sweep include brooms, dustpans and mops."*	*"Verbs that are semantically close to sweep include cleane, purify, and eradicate."*

transform them along with other sensory inputs into their feature representations. Therefore, two encoders are introduced: one for embedding point cloud data and the other for embedding language data.

Object and Grasp Encoder To model the relative spatial relationship between a 6 DoF (Degree of Freedom) grasp pose and X_o, we adopt a joint embedding strategy. Following Mousavian et al. [21], we first approximate the robot gripper with six control points X_g defined in the object frame and concatenate them to the object point cloud X_o to form a joint point cloud. A binary feature vector is then added to the joint point cloud, indicating that each point belongs to the object or the gripper. Finally, the joint point cloud is embedded with PointNet++ [26] (denoted as PN++), which consists of three set abstraction layers:

$$F_X = \text{PN++}(\text{Concat}([X_g, X_o], \text{dim} = 0))$$

The resulting point cloud embedding $F_X \in \mathbb{R}^{1024}$ is later fused with language embeddings.

Language Encoder In order to relate known and novel concepts, GraspGPT necessitates the ability to digest a large variety of linguistic elements in language descriptions. For instance, in the case of an affordance description of the task *"pour":*

"*Household objects that support the function of pouring include utensils such as pitchers, cups, and ladles, as well as containers with pouring spouts, to aid in the transfer of liquid or other items from one vessel to another.*"

GraspGPT must be able to comprehend object classes (e.g., *pitchers*, *cups*, *ladles*), entity taxonomy (e.g., *utensil*, *container*, *vessel*, *liquid*), actions (e.g., *transfer*), and relations (e.g., *from... to...*). While training a dedicated language encoder from scratch is a common choice, it would require a significant amount of training data and is meanwhile time-consuming. We, therefore, opt for a BERT [27] pre-trained on a large corpus of text data to encode both language descriptions and instructions. The pre-trained BERT outputs word embeddings for a task description paragraph $F_{td} \in \mathbb{R}^{T_{td} \times 768}$, an object class description paragraph $F_{od} \in \mathbb{R}^{T_{od} \times 768}$, and a language instruction $F_I \in \mathbb{R}^{T_I \times 768}$, where T_{td}, T_{od}, and T_I denote the maximum lengths (with zero-padding) for each language sequence, respectively. The language encoder is frozen during training.

D. Task-Oriented Grasp Evaluation

After obtaining the feature representations of all elements, we next present a multi-modal fusion module for task-oriented grasp evaluation, which can be represented below:

$$S = \text{TGE}(F_X, F_I, F_{td}, F_{od})$$

where TGE is the task-oriented grasp evaluator, and S is the score for the candidate grasp pose g.

Task-Oriented Grasp Evaluator TGE is implemented as a customized Transformer decoder [28]. It is analogous to a sequence-to-sequence model commonly used in machine translation, which converts sequences from one domain to another. In our problem, the robot is unable to comprehend novel concepts out of the training set. We utilize TGE to translate a novel concept using its description paragraphs, and connect the novel concept to its related concepts described during training.

The architecture of TGE is depicted in Fig. 2(b). The translation process incorporates contextual information from language descriptions into their corresponding concept in I. Both training and inference follow the same computational procedure. Specifically, we begin by transforming word embeddings from the pre-trained language encoder to a lower dimension space and obtain $\tilde{F}_{td} \in \mathbb{R}^{T_{td} \times 128}$, $\tilde{F}_{od} \in \mathbb{R}^{T_{od} \times 128}$, and $\tilde{F}_I \in \mathbb{R}^{T_I \times 128}$. TGE consists of two layers, one for incorporating object class knowledge $\tilde{F}_{od}$ and the other for incorporating task knowledge $\tilde{F}_{td}$. Each decoder layer aims to learn a function as below:

$$\phi_{td} : \mathbb{R}^{T_I \times 128} \times \mathbb{R}^{T_{td} \times 128} \rightarrow \mathbb{R}^{T_I \times 128}$$

$$\phi_{od} : \mathbb{R}^{T_I \times 128} \times \mathbb{R}^{T_{od} \times 128} \rightarrow \mathbb{R}^{T_I \times 128}$$

where the outputs of ϕ_{td} and ϕ_{od} are language instruction word embeddings augmented with contextual knowledge. Two decoder layers share a similar design. The computational procedure of ϕ_{*d} can be represented as follows:

$$\tilde{F}_I = \text{LN}(\tilde{F}_I + \text{MHA}(\tilde{F}_I, \tilde{F}_{*d}))$$

$$\tilde{F}_I = \text{LN}(\tilde{F}_I + \text{FFN}(\tilde{F}_I))$$

where $*d$ can be either td or od; LN, MHA, and FFN denote layer normalization, multi-head attention, and feedforward network, respectively; MHA consists of eight cross-attention heads in our implementation. The computation of each cross-attention head can be represented as:

$$A = \text{Softmax}\left(\frac{Q_I K_{*d}^T}{\sqrt{128}}\right) V_{*d}$$

where A is the attended word embeddings. Q_I, K_{*d}, and V_{*d} are transformed from $\tilde{F}_I$ and $\tilde{F}_{*d}$ as follows:

$$Q_I = \text{Q}_{\text{proj}}(\tilde{F}_I), K_{*d} = \text{K}_{\text{proj}}(\tilde{F}_{*d}), V_{*d} = \text{V}_{\text{proj}}(\tilde{F}_{*d})$$

where Q_{proj}, K_{proj}, and V_{proj} are projection matrices. The intuition is to reconstruct $\tilde{F}_I$ by all elements in $\tilde{F}_{*d}$ weighted by their normalized correspondence. Since cross-attention mechanism

can dynamically assign weights to each input token, it learns to attend to concept tokens in I while ignore irrelevant tokens.

Finally, $\tilde{F}_I$ is mean pooled to output a sentence embedding $\overline{F}_I \in \mathbb{R}^{128}$. It is then concatenated with the shape embedding $\tilde{F}_X \in \mathbb{R}^{300}$, which is obtained by projecting F_X via a fully connected layer. We compute S using an MLP (Multi-Layer Perceptron) with a sigmoid activation:

$$S = \text{Sigmoid}(\text{MLP}(\text{Concat}([\tilde{F}_X, \overline{F}_I], \text{dim} = -1)))$$

The MLP comprises three fully connected layers with 1D batch normalization, ReLU activation, and dropout.

Loss Function We compute the binary cross-entropy loss between S and the ground truth label S_{gt}:

$$\mathcal{L}_{bce} = -\frac{1}{N}\sum_{i=1}^{N} S_{gt,i} \cdot \log(S_i) + (1 - S_{gt,i}) \cdot \log(1 - S_i)$$

where N is the total number of samples, and $S_{gt,i}$ is set to one if the i_{th} grasp pose is successful and zero otherwise.

V. Experimental Setup

A. Perception Experiments

Baselines We compare GraspGPT to the following methods: 1) **Random**, which represents the method in [22] that focuses on grasp stability only and ignores task constraints (i.e., task-agnostic grasping method). 2) **Semantic Grasp Network (SGN)** [12], which learns class-task-grasp relations without incorporating external semantic knowledge. 3) **GCNGrasp** [4], which is the state-of-the-art TOG algorithm introduced earlier and whose main limitation is its inability to generalize to novel concepts out of the graph. During inference, we connect the novel concept node to its nearest neighbor in the KG. The nearest neighbor search is based on the cosine similarity between the concepts' pre-trained word embeddings provided by ConceptNet [29].

Dataset GraspGPT and three baselines are evaluated on the LA-TaskGrasp dataset, which augments the TaskGrasp dataset with language data. The original TaskGrasp dataset contains 250 K task-oriented grasp pose annotations for 56 tasks, 75 object classes, and 191 object instances. Each instance is a partial point cloud of a real household object **with multi-view RGB-D fusion**. TaskGrasp provides three types of held-out settings: held-out (object) class, held-out task, and held-out instance. We focus on the former two settings in this letter. For language data, LA-TaskGrasp contains 80 language description records for each object class and 40 records for each task, resulting in 6000 object class description records and 2240 task description records. We combine these descriptions to generate 750 object class description paragraphs and 560 task description paragraphs. LA-TaskGrasp dataset also includes 53 language instruction templates, resulting in 222600 possible language instruction sentences.

Metrics We use the same set of evaluation metrics used by GCNGrasp. Specifically, we compute the Average Precision (AP) score for each object class, task, and instance, and then compute the mean AP (mAP) averaged over all object classes, tasks, and instances (i.e., class mAP, task mAP, and instance mAP).

B. Real-Robot Experiments

The real-robot experiment platform comprises a 7 DoF Kinova Gen3 robotic arm with a parallel jaw gripper and an Intel RealSense D435 RGB-D camera with eye-in-hand calibration. For each test object, we first apply SAM [30] to extract the object point cloud **captured from a single view** and then apply Contact-GraspNet [22] to generate 50 grasp pose candidates. Single-view setup is used here because it is more practical for real-world robotic applications. Finally, all the candidates are evaluated and the one with the highest score are executed. We collect test objects from our laboratory and YCB dataset. More details on the experimental setup can be found in the appendix.

The physical grasping pipeline is divided into three stages: Perception, Planning, and Action, and the statistics of each stage is reported separately for clarity. A trial succeeds if the test object is grasped subject to the natural language instruction and lifted stably by the robot. We additionally combine GraspGPT with three pre-defined skills (pouring, handover, and scooping) in the form of motion primitive to showcase its practicality in tool manipulation.

C. Implementation Details

All the experiments are conducted on a desktop PC with a single Nvidia RTX 3090 GPU. GraspGPT is optimized with an Adam optimizer [31] with a weight decay of 0.0001. The learning rate is set to 0.0001 initially and decays subject to a customized function as in GCNGrasp. We train GraspGPT for 50 epochs with a batch size of 32. Each point cloud is downsampled to 4096 points before being fed into the model.

For the choice of an LLM, we select the OpenAI GPT-3 model, specifically the *text-davinci-003* version. GraspGPT is capable of incorporating any current LLM, such as OpenAI GPT-4 and Google Bard, or using an ensemble of LLMs. We leave this ensemble approach for our future work. For the language encoder, we choose the Google pre-trained *BERT-Base* model provided by Hugging Face.

VI. Results

A. Results of Perception Experiments

To highlight the difference between our approach and GCNGrasp, we investigate perception experiments from two perspectives: open-world generalization and closed-world generalization. The former evaluates the generalization performance to novel concepts out of the knowledge graph of GCNGrasp. In the latter evaluation, GCNGrasp has access to all the concepts in the LA-TaskGrasp dataset and the ground truth relations between them in its pre-defined graph. Although this assumption is impractical in real-world robotic applications, we still want to explore how our approach compares to GCNGrasp even though GraspGPT does not assume access to concepts out of the training set. Since GraspGPT and the other two baselines do not rely on a pre-defined graph structure, their results for the two evaluations are the same. The quantitative results of perception experiments are reported in Table II.

Open-World Generalization For both held-out settings, Random achieves approximate mAPs of 50–60%, indicating that the distribution of positive and negative samples in the dataset is even. By considering task constraints, SGN achieves consistent improvements (10%+) over Random under two held-out

TABLE II
QUANTITATIVE RESULTS OF PERCEPTION EXPERIMENTS

Method	Held-out Class Performance (mAP)			Held-out Task Performance (mAP)		
	Instance	Class	Task	Instance	Class	Task
Random	59.32	58.73	52.72	59.06	58.24	52.37
SGN	74.20	72.95	62.55	75.17	71.59	63.35
GCNGrasp (open-world)	72.92	72.45	67.58	57.48	47.21	33.17
GCNGrasp (closed-world)	79.35	76.88	72.97	80.43	76.06	76.11
GraspGPT (full model)	79.70	77.88	72.84	79.32	76.90	72.34
GraspGPT (w/o D)	74.10	74.33	66.38	74.66	70.85	68.14
GraspGPT (w/o TD)	80.95	77.74	73.76	75.00	71.21	68.38
GraspGPT (w/o OD)	76.04	74.71	71.60	78.26	74.71	71.60

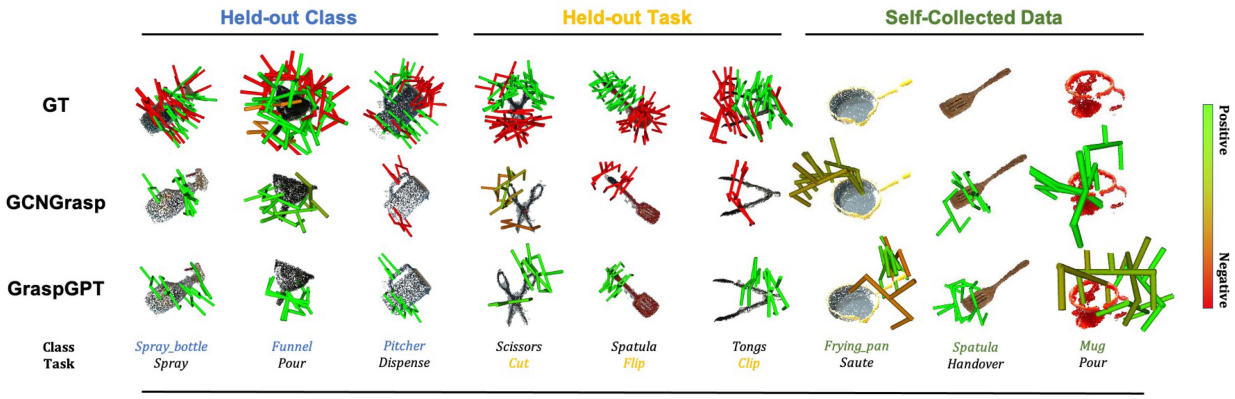

Fig. 3. Qualitative results of open-world generalization. GraspGPT and GCNGrasp are evaluated under both held-out settings. Results on self-collected test objects (no ground truth annotations) are also presented. Grasp poses are colored by their confidence scores (green is higher). Only top-5 predictions are displayed for better visualization effect.

settings. For GCNGrasp, we observe a significant performance difference between the held-out task setting and the held-out class setting. GCNGrasp even falls behind Random by 1.58%, 11.03%, and 19.20% on three metrics in the held-out task setting. This suggests that the pre-trained word embeddings of ConceptNet are good at capturing the linguistic relations between object classes but perform poorly on relating task concepts. Therefore, GCNGrasp cannot fully exploit the power of semantic knowledge encoded in its graph. Since GraspGPT does not rely on a pre-defined KG but instead leverages the open-end semantic knowledge from an LLM, **GraspGPT outperforms all three baselines when generalizing to concepts out of the training set**. It outperforms GCNGrasp by 21.84%, 29.69%, and 39.17% on held-out task setting and by 6.78%, 5.43%, and 5.26% on held-out class setting. The qualitative results are presented in Fig. 3.

Closed-World Generalization Compared to open-world generalization, GCNGrasp performs consistently better on closed-world generalization since all the concepts and the ground truth relations between them have been pre-defined in its graph. GraspGPT and GCNGrasp outperform both Random and SGN due to the incorporation of semantic knowledge. For the held-out task setting, GraspGPT achieves comparable performance with GCNGrasp on instance mAP and class mAP but falls behind by 3.77% on task mAP. For held-out class setting, GraspGPT outperforms GCNGrasp on two metrics. Overall, **GraspGPT achieves comparable performance with GCNGrasp on closed-world generalization** even though it does not assume access to all concepts and their relations as GCNGrasp does.

B. Results of Real-Robot Experiments

Task-Oriented Grasping We conduct 100 trials on each held-out setting, with ten trials per object class or task. As presented in Table III, GraspGPT achieves high success rates (86.00% and 91.00%) in the perception stage, even though the object point clouds are captured from a single view. The performance drop from the perception stage to the action stage (71.00% and 77.00%) can be attributed to three primary reasons: 1) marginal grasp candidates generated by the grasp sampler; 2) incorrect evaluation by GraspGPT; 3) motion planning failure. The qualitative results of three test objects are shown in Fig. 3 (right).

Task-Oriented Manipulation To support task-oriented manipulation (refer to Fig. 4), we first utilize GraspGPT to generate task-oriented grasp poses for tool objects. Then, we design rule-based heuristics to determine the operating direction and effect points [32] on the target objects. As presented in Table IV, GraspGPT performs well in task-oriented grasping, achieving

TABLE III
RESULTS OF TASK-ORIENTED GRASPING EXPERIMENTS

Method	Held-out Class Performance			Success	Held-out Task Performance			Success
	Perception	Planning	Action		Perception	Planning	Action	
GraspGPT	91/100	85/100	77/100	77.00%	86/100	77/100	71/100	71.00%

TABLE IV
RESULTS OF TASK-ORIENTED MANIPULATION EXPERIMENTS

Method	Pouring		Success	Handover		Success	Scooping		Success
	Grasping	Manipulation		Grasping	Manipulation		Grasping	Manipulation	
GraspGPT	15/20	12/20	60.00%	17/20	16/20	80.00%	18/20	13/20	65.00%

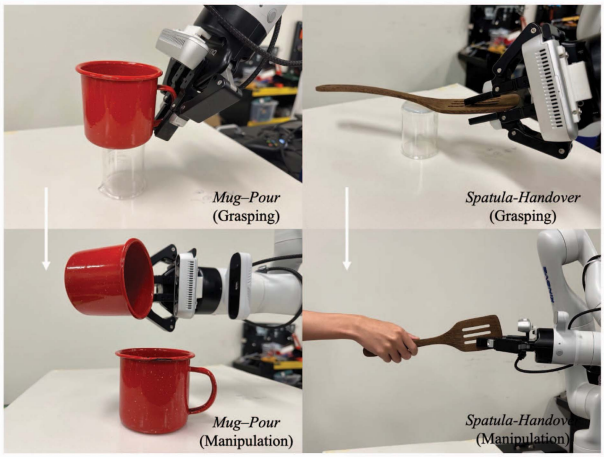

Fig. 4. Real-robot experiments on task-oriented grasping and manipulation: *Mug-Pour* (left) and *Spatula-Handover* (right).

success rates of 75.00%, 85.00%, and 90.00% in three tasks, respectively. However, due to its inability to adaptively model the relative pose [33] between the tool object and the target object, the success rates of task-oriented manipulation decrease, especially for pouring and scooping. Future work includes extending GraspGPT to support task-oriented pick and place.

C. Ablation Study

To gain further insights into the effectiveness of each component of GraspGPT, we perform two sets of ablation studies, aiming to answer two questions:

- Does the incorporation of semantic knowledge from an LLM help to better generalize to novel concepts out of the training set?
- How does the selection of a pre-trained language encoder affect the overall performance of GraspGPT?

Ablation on Semantic Knowledge We compare GraspGPT to three ablations: 1) no semantic knowledge (i.e., w/o D); 2) object class description only (i.e., w/o TD); 3) task description only (i.e., w/o OD). The results for two held-out settings are reported in Table II. For the held-out task setting, the full model outperforms all three ablations. Specifically, the comparison between w/o D - w/o TD and w/o D - w/o OD demonstrates that the incorporation of task knowledge is more important for novel task generalization. For the held-out class setting, we observe that object class knowledge is more important for generalizing to novel classes out of the training set. Using object class knowledge only (w/o TD) even slightly outperforms the full model. We argue that object class descriptions have already provided sufficient knowledge for novel object class generalization. Arbitrarily incorporating extra task knowledge may lead to adversarial/conflicting effects in some cases. In our current implementation, we do not preprocess the language data from the LLM. Future work will be done on knowledge filtering and selection. Overall, the result verifies the hypothesis that **the incorporation of semantic knowledge helps achieve better generalization to novel concepts.**

TABLE V
ABLATION ON PRE-TRAINED LANGUAGE ENCODER

Model	Held-out Task Performance (mAP)		
	Instance	Class	Task
BERT-Small (L=4)	78.06	74.48	71.90
BERT-Medium (L=8)	78.47	75.93	72.02
BERT-Base (L=12)	**79.32**	**76.06**	**72.34**

Ablation on Language Encoder To validate the design choice of using a large pre-trained language encoder, we equip GraspGPT with pre-trained BERTs of three sizes and compare their resulting mAPs. Since the conclusions for the two held-out settings are similar, we only report the result of the held-out task setting for simplicity. The result is presented in Table V, where L denotes the number of transformer layers. It is clear that *BERT-Base* **outperforms two smaller models**, but the gaps are insignificant. We argue that the three models are equally pre-trained on a large corpus of text data, so they achieve a similar level of knowledge understanding capability despite their differences in model complexity.

VII. CONCLUSION

In this letter, we propose GraspGPT, an LLM-based TOG framework that leverages the open-end semantic knowledge from an LLM to achieve zero-shot generalization to novel concepts out of the training set. Compared to existing methods, GraspGPT does not rely on any pre-defined concept set or

knowledge base. Evaluation on the LA-TaskGrasp dataset demonstrates the superiority of GraspGPT over existing methods on novel concept generalization. The effectiveness of GraspGPT is further validated in performing task-oriented grasping and manipulation in real-world applications.

REFERENCES

[1] M. Kokic, D. Kragic, and J. Bohg, "Learning task-oriented grasping from human activity datasets," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 3352–3359, Apr. 2020.

[2] K. Fang et al., "Learning task-oriented grasping for tool manipulation from simulated self-supervision," *Int. J. Robot. Res.*, vol. 39, no. 2/3, pp. 202–216, 2020.

[3] D. Song, K. Huebner, V. Kyrki, and D. Kragic, "Learning task constraints for robot grasping using graphical models," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 1579–1585.

[4] A. Murali, W. Liu, K. Marino, S. Chernova, and A. Gupta, "Same object, different grasps: Data and semantic knowledge for task-oriented grasping," in *Proc. Conf. Robot Learn.*, 2021, pp. 1540–1557.

[5] A. Chowdhery et al., "Palm: Scaling language modeling with pathways," 2022, *arXiv:2204.02311*.

[6] R. Thoppilan et al., "Lamda: Language models for dialog applications," 2022, *arXiv:2201.08239*.

[7] M. Ahn et al., "Do as I can, not as I say: Grounding language in robotic affordances," 2022, *arXiv:2204.01691*.

[8] A. Z. Ren, B. Govil, T.-Y. Yang, K. R. Narasimhan, and A. Majumdar, "Leveraging language for accelerated learning of tool manipulation," in *Proc. Conf. Robot Learn.*, 2023, pp. 1531–1541.

[9] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *Proc. Int. Conf. Mach. Learn.,* 2022, pp. 9118–9147.

[10] J. Liang et al., "Code as policies: Language model programs for embodied control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 9493–9500.

[11] H. Dang and P. K. Allen, "Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 1311–1317.

[12] W. Liu, A. Daruna, and S. Chernova, "CAGE: Context-aware grasping engine," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 2550–2556.

[13] P. Ardó n, É. Pairet, R. P. A. Petrick, S. Ramamoorthy, and K. S. Lohan, "Learning grasp affordance reasoning through semantic relations," *IEEE Robot. Automat. Lett.*, vol. 4, no. 4, pp. 4571–4578, Oct. 2019.

[14] L. Antanas et al., "Semantic and geometric reasoning for robotic grasping: A probabilistic logic approach," *Auton. Robots*, vol. 43, pp. 1393–1418, 2019.

[15] D. Shah, B. Osiński, and and S. Levine, "LM-NAV: Robotic navigation with large pre-trained models of language, vision, and action," in *Proc. Conf. Robot Learn.*, 2023, pp. 492–504.

[16] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual language maps for robot navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022,pp. 10608–10615.

[17] J. Wu et al., "TidyBot: Personalized robot assistance with large language models," 2023, *arXiv:2305.05658*.

[18] W. Huang et al., "Inner monologue: Embodied reasoning through planning with language models," in *Proc. Conf. Robot. Learn. PMLR*, 2023, pp. 1769–1782.

[19] Y. Jiang et al., "VIMA: General robot manipulation with multimodal prompts," 2022, *arXiv:2210.03094*.

[20] A. Zeng et al., "Socratic models: Composing zero-shot multimodal reasoning with language," in *Proc. 11th Int. Conf. Learn. Representations*, 2022.

[21] A. Mousavian, C. Eppner, and D. Fox, "6-DOF GraspNet: Variational grasp generation for object manipulation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 2901–2910.

[22] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox, "Contact-GraspNet: Efficient 6-DoF grasp generation in cluttered scenes," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 13438–13444.

[23] E. Rosch, "Cognitive representations of semantic categories," *J. Exp. Psychol.: Gen.*, vol. 104, no. 3, 1975, Art. no. 192.

[24] C. Tang, D. Huang, L. Meng, W. Liu, and H. Zhang, "Task-oriented grasp prediction with visual-language inputs," 2023, *arXiv:2302.14355*.

[25] T. Nguyen, N. Gopalan, R. Patel, M. Corsaro, E. Pavlick, and S. Tellex, "Affordance-based robot object retrieval," *Auton. Robots*, vol. 46, no. 1, pp. 83–98, 2022.

[26] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++ : Deep hierarchical feature learning on point sets in a metric space," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5105–5114.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[28] A. Vaswani et al., "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[29] H. Liu and P. Singh, "ConceptNet-a practical commonsense reasoning tool-kit," *BT Technol. J.*, vol. 22, no. 4, pp. 211–226, 2004.

[30] A. Kirillov et al., "Segment anything," 2023, *arXiv:2304.02643*.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[32] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese, "KETO: Learning keypoint representations for tool manipulation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 7278–7285.

[33] C. Pan, B. Okorn, H. Zhang, B. Eisner, and D. Held, "TAX-Pose: Task-specific cross-pose estimation for robot manipulation," in *Proc. Conf. Robot Learn.*, 2023, pp. 1783–1792.