Robots That Ask For Help: Uncertainty Alignment for Large Language Model Planners

Anonymous Author(s)
Affiliation
Address
email

Abstract: Large language models (LLMs) exhibit a wide range of promising capabilities — from step-by-step planning to commonsense reasoning — that may provide utility for robots, but remain prone to *confidently* hallucinated predictions. In this work, we present KNOWNO, a framework for measuring and aligning the uncertainty of LLM-based planners, such that they know when they don't know, and ask for help when needed. KNOWNO builds on the theory of conformal prediction to provide statistical guarantees on task completion while minimizing human help in complex multi-step planning settings. Experiments across a variety of simulated and real robot setups that involve tasks with different modes of ambiguity (e.g., from spatial to numeric uncertainties, from human preferences to Winograd schemas) show that KNOWNO performs favorably over modern baselines (which may involve ensembles or extensive prompt tuning) in terms of improving efficiency and autonomy, while providing formal assurances. KNOWNO can be used with LLMs out-of-the-box without model-finetuning, and suggests a promising lightweight approach to modeling uncertainty that can complement and scale with the growing capabilities of foundation models.[1]

Keywords: Language-based planning, uncertainty estimation, conformal prediction

1 Introduction

How can we endow our robots with the ability to *know when they don't know*? Accurately modeling and accounting for uncertainty is a longstanding challenge towards robots that operate reliably in unstructured and novel environments. In this work, we study this challenge in the context of *language-instructed* robots. Language provides a natural and flexible interface for humans to specify tasks, contextual information, and intentions, while also allowing us to provide help and clarification to robots when they are uncertain.

Recently, approaches that leverage large language models (LLMs) for planning [1, 2] have demonstrated the ability to respond to natural and unstructured language instructions to generate temporally extended plans. These approaches enable leveraging the vast amount of prior knowledge and rich context embedded in pretrained LLMs, and lead to substantial abstract reasoning capabilities. However, one of the major challenges with current LLMs is their tendency to *hallucinate*, i.e., to *confidently* generate outputs that are plausible but incorrect and untethered from reality. Such false confidence in incorrect outputs poses a significant challenge to LLM-based planning in robotics. Natural language instructions in real-world environments often contain a high degree of inherent ambiguity, and confidently following an incorrectly constructed plan could lead to undesirable or even unsafe actions.

As an example, a robot tasked with heating food may choose to place a metal bowl in the microwave, potentially damaging the microwave or even causing a fire (Fig. 1). Instead of following such a plan, we would like our robots to know when they don't know and ask for clarification instead (e.g., ask which bowl is safe to be placed in the microwave). Prior work in language-based planning either does not seek such clarifications [1] or does so via extensive prompting [2], which requires careful prompt engineering to prevent the robot from excessively relying on seeking assistance. Moreover, prior approaches do not provide a way to ensure that asking for help results in a desired level of task success. We formalize

[1] Webpage with additional results and videos: https://tinyurl.com/robot-help

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

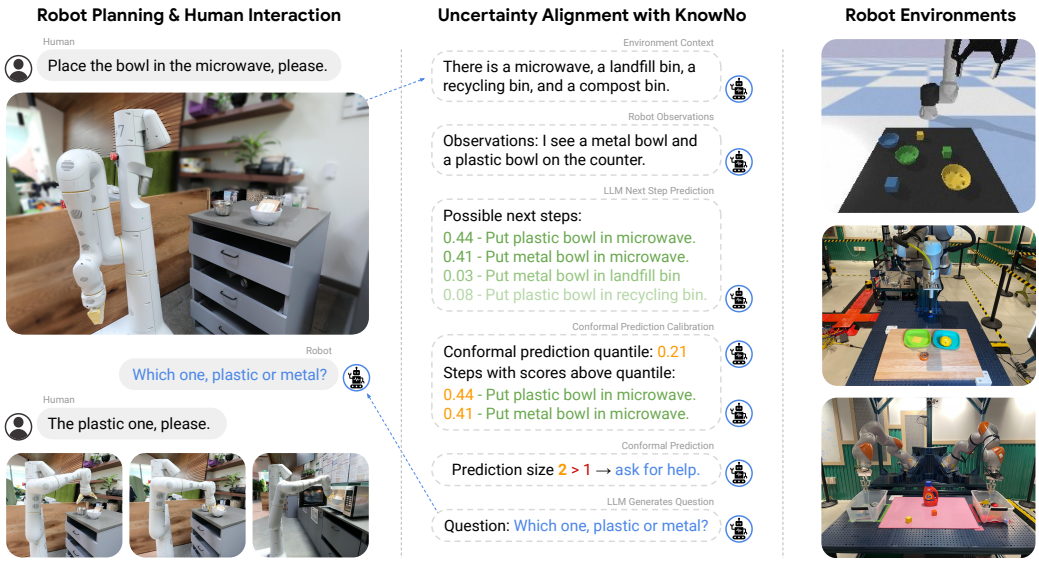

Figure 1: KNOWNO uses Conformal Prediction (CP) to align the uncertainty of LLM planners. Given language instructions, an LLM generates a set of possible next steps (with scores above a certain CP calibration quantile). If there are more than one in the set, the robot asks for help, then readjusts accordingly. Experiments show that KNOWNO works well with multiple embodiments, across a variety of ambiguous situations, in both simulated and real setups.

these challenges via two desiderata: (i) *calibrated confidence:* the robot should seek sufficient help to ensure a statistically guaranteed level of task success specified by the user, and (ii) *minimal help:* the robot should minimize the overall amount of help it seeks by narrowing down possible ambiguities in a task. We collectively refer to these sufficiency and minimality conditions as *uncertainty alignment*.

Statement of contributions. We propose KNOWNO— *Know When You Don't Know* — a framework for aligning the uncertainty of LLM-based planners utilizing the theory of *conformal prediction (CP)* [3, 4]. We make the following contributions: **(1)** Given a language instruction, we utilize a pre-trained LLM with uncalibrated confidence to generate a *set* of possible actions for the robot to execute next. We demonstrate how to use CP to select a subset of these options, which allows the robot to decide an action to execute (if the subset is a singleton) or to ask for help otherwise. **(2)** We prove theoretical guarantees on calibrated confidence in both single-step and multi-step planning problems: with a user-defined confidence $1-\epsilon$ over instructions, the robot performs all actions for a given instruction correctly by asking for help when necessary. CP also minimizes the average size of prediction sets, thus addressing the goal of minimal help. **(3)** We evaluate KNOWNO in both simulation and hardware with a suite of language-instructed manipulation tasks with various types of potential ambiguities (e.g., based on spatial locations, numerical values, attributes of objects, and Winograd schemas). Experiments across multiple settings and embodiments validate the ability of KNOWNO to provide statistically guaranteed levels of task success while significantly reducing the amount of help required to achieve a given success level by $10-15\%$ as compared to baseline approaches.

2 Overview: Robots that Ask for Help

Language-based planners. Language model planners can generate step-by-step robot plans, where each step y is composed of variable-length sequences of symbols $(\sigma_1,\sigma_2,...,\sigma_k)$, e.g., text tokens as input to a language-conditioned policy [1] (see Fig. 1), or robot code executed by an interpreter [5]. Pretrained autoregressive LLMs predict each step y, whose joint probability over tokens can be factorized as the product of conditional probabilities of next token prediction $p(y) = \prod_{i=1}^{k} p(\sigma_i \mid \sigma_1,...,\sigma_{i-1})$. Here, we are interested in characterizing the uncertainty of next step prediction $p(y)$. The distribution of $p(y)$ values (i.e., perplexity) remains highly sensitive to variable-length k; hence $p(y)$ on its own serves as a rather poor scoring function [6] particularly when steps in a plan are expressed in natural language (our experiments in Section A8 also show that using $p(y)$ directly for calibration leads to poor performance).

Planning as multiple-choice Q&A. We can address this length bias with a simple trick, whereby we can prompt the LLM to generate (either with high-temperature sampling, or with another LLM instance)

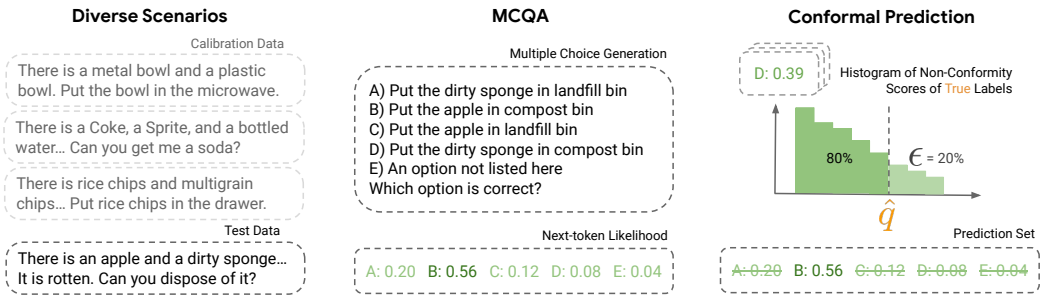

Figure 2: KNOWNO formulates LLM planning as MCQA by first prompting LLM to generate plausible options, then asking it to predict the correct one. Using the next-token likelihoods collected with a calibration dataset, CP provides the quantile value $\hat{q}$ such that all options with a score $\geq 1-\hat{q}$ are included in the prediction set in a test scenario.

a *set* $\{y^i\}$ of candidate next steps (e.g., "Put plastic bowl in microwave", "Put metal bowl in microwave", etc., in Fig. 1), and then format the task of choosing among them as multiple-choice Q&A (MCQA). This eliminates plans that the LLM considers unlikely and reduces the problem of next-step prediction down to a single next-token prediction — aligning with LLM log-likelihood loss functions and LLM training data (e.g., MCQA datasets [7, 8]). These probabilities can serve as normalized scores that can be used by various uncertainty quantification methods such as thresholding, ensemble methods, and others. In this work, we use these normalized scores within a conformal prediction (CP) framework. Specifically, CP uses a held-out calibration set of example plans in different scenarios to generate a reduced prediction set of plans among $\{y^i\}$ (Fig. 2). The LLM is certain if this prediction set is a singleton, and triggers help from a human otherwise.

Robots that ask for help. In this work, we show that LLM planning — combined with CP for uncertainty estimation — can effectively enable robots to interact with an environment, and ask for help when needed. The environment e can be formulated as a partially observable Markov decision process (POMDP): at any given state s_t at time t, given a user instruction ℓ, the robot executes an action a_t according to a policy π, then transitions to a new state s_{t+1}. Our policy π is composed of four parts (Fig. 1):

1. Multiple-choice generation: An LLM generates a diverse set of candidate plans labeled with 'A', 'B', 'C', 'D' , and an additional possible plan, 'E) an option not listed here', which is appended post-hoc. We denote the set of labels by $\mathcal{Y} := \{$'A',' B',' C',' D',' E'$\}$. These plans are generated by prompting the LLM with *context* x^t, which is text that (1) summarizes the robot observation at each time step (e.g., using a vision-based object detector or an oracle; see Fig. 1) and (2) includes the user instruction. An *augmented context* $\tilde{x}^t$ is obtained by appending the LLM-generated plans to the context x^t.
2. Prediction set generation: We use CP to choose a *subset* $C(\tilde{x}^t) \subseteq \mathcal{Y}$ of candidate plans using the LLM's (uncalibrated) confidence $\hat{f}(\tilde{x}^t)_y$ in each prediction $y \in \mathcal{Y}$ given the context $\tilde{x}^t$.
3. Human help: If the prediction set is a non-singleton, the robot leverages help from a human (or any other supervisor agent, denoted as a function $f_{\mathcal{H}}$) to arrive at an unambiguous next step $y_{\mathcal{H}} \in C(\tilde{x}^t)$.
4. Low-level control: A low-level module φ converts the plan in $y_{\mathcal{H}}$ to an action $a_t = \varphi(y_{\mathcal{H}})$.

Goal: uncertainty alignment. Often in real-world settings, language instructions ℓ can be ambiguous, e.g., "place the bowl in the microwave" does not specify that the human may prefer the plastic bowl (Fig. 1). Our goal in this work is to address uncertainty alignment: achieve a desired level of task success while minimizing human help. We formalize this by considering a joint distribution $\mathcal{D}$ over *scenarios* $\xi := (e, \ell, g)$, where e is an environment (POMDP), ℓ is a (potentially ambiguous) language instruction, and g is a goal (e.g., formulated as a subset of acceptable states in the POMDP). Importantly, we do not assume knowledge of $\mathcal{D}$, except that we can sample a finite-size dataset of i.i.d. scenarios from it. We formalize uncertainty alignment in our setting as (i) *calibrated confidence:* the robot's policy (with human help as described above) succeeds with a user-specified probability $1-\epsilon$ over new scenarios $\xi \sim \mathcal{D}$, and (ii) *minimal help:* the policy minimizes the number $|C(\cdot)|$ of options presented to the human on average across scenarios $\xi \sim \mathcal{D}$.

3 Calibrating LLM Confidence with Conformal Prediction

The MCQA setup above allows us to apply CP to obtain calibrated confidence guarantees while (approximately) minimizing help. We introduce CP below, and then present the different practical settings we consider (possibly involving multiple planning steps and/or multiple correct plans per step).

3.1 Background: Conformal Prediction

For now, we drop the timestep superscript and consider a generic MCQA setup with pairs $(\tilde{x},y)$ consisting of input $\tilde{x}$ and true label y. Suppose there is a *calibration* set $Z=\{z_i=(\tilde{x}_i,y_i)\}_{i=1}^N$ of such pairs drawn i.i.d. from an *unknown* distribution $\mathcal{D}$ over $\mathcal{Z}:=\mathcal{X}\times\mathcal{Y}$. Now, given a new i.i.d. sample $z_{\text{test}}=(\tilde{x}_{\text{test}},y_{\text{test}})$ with unknown true label y_{test}, CP generates a *prediction set* $C(\tilde{x}_{\text{test}})\subseteq\mathcal{Y}$ that contains y_{test} with high probability [3]:

$$\mathbb{P}\big(y_{\text{test}}\in C(\tilde{x}_{\text{test}})\big)\geq 1-\epsilon, \tag{1}$$

where $1-\epsilon$ is a user-defined threshold (desired task success level in our setting) that affects the size of $C(\cdot)$.

To generate $C(\tilde{x}_{\text{test}})$, CP first uses the LLM's confidence $\hat{f}$ (cf. Section 2) to evaluate the set of *nonconformity scores* $\{s_i=1-\hat{f}(\tilde{x}_i)_{y_i}\}_{i=1}^N$ over the calibration set — the higher the score is, the less each data in the calibration set *conforms* to the data used for training $\hat{f}$. Then CP performs calibration by defining $\hat{q}$ to be the $\frac{\lceil(N+1)(1-\epsilon)\rceil}{N}$ empirical quantile of $s_1,\ldots,s_N$. Lastly, CP generates $C(\tilde{x}_{\text{test}})=\{y\in\mathcal{Y}\mid\hat{f}(\tilde{x}_{\text{test}})_y\geq 1-\hat{q})\}$, i.e., the prediction set that includes all labels that the predictor is at least $1-\hat{q}$ confident in. The generated prediction set ensures that the coverage guarantee in Eq. (1) holds.

Dataset-conditional guarantee. The probability in Eq. (1) is over *both* the sampling of the calibration set Z and z_{test} (i.e., a *marginal* guarantee). Thus, to ensure the desired probability of coverage for each new z_{test}, one needs a fresh calibration set. But, in practice, we only calibrate once with a fixed set. The following *dataset-conditional* guarantee [9] holds with probability $1-\delta$ over the sampling of the calibration set Z:

$$\mathbb{P}\big(y_{\text{test}}\in C(\tilde{x}_{\text{test}})\mid\{z_1,\ldots,z_N\}\big)\geq\text{Beta}^{-1}_{N+1-v,v}(\delta),\quad v:=\lfloor(N+1)\hat{\epsilon}\rfloor, \tag{2}$$

where $\text{Beta}^{-1}_{N+1-v,v}(\delta)$ denotes the inverse CDF (quantile) level of δ in a Beta distribution with parameters $N+1-v$ and v, and $\hat{\epsilon}$ is the threshold used for calibration. In practice, we use a modest-sized calibration dataset ($N=400$) and $\delta=0.01$, and adjust $\hat{\epsilon}$ to achieve the desired $1-\epsilon$ coverage (with probability $1-\delta=0.99$ over the sampling of the calibration set).

Minimal prediction set size. From [10, Thm. 1], $C(\cdot)$ achieves the smallest average set size among possible prediction schemes $\mathcal{C}$ that achieve the coverage guarantee, if $\hat{f}(\tilde{x})_y$ models true conditional probabilities:

$$\min_{C\in\mathcal{C}}\ \mathop{\mathbb{E}}_{(\tilde{x},\cdot)\sim\mathcal{D}}\big[|C(\tilde{x})|\big],\ \text{subject to (1)}. \tag{3}$$

The assumption that $\hat{f}$ models true conditional probabilities may be a good approximation for LLMs trained on large-scale data with a proper scoring rule [11]; one can also obtain bounds on near-optimal average set sizes for CP using $\hat{f}$ that approximately model conditional probabilities [12, 10], but we omit these results for brevity. Overall, CP is a powerful and easy-to-use statistical tool to produce (1) tight coverage guarantees—addressing the goal of *calibrated confidence* and (2) small prediction sets for unseen data given a blackbox predictor like a LLM and an unknown data distribution—addressing our second goal of *minimal help*.

3.2 Single-Step Uncertainty Alignment

We now demonstrate how to use CP to achieve uncertainty alignment with a user-defined task completion rate $1-\epsilon$. We first consider a single-step setting, where the LLM plans only once given a context. For simplicity, we again drop the timestep superscript t in this section.

Data collection. We collect N i.i.d. scenarios from the distribution $\mathcal{D}$, and the corresponding contexts summarizing the robot observation and instruction. We use the MCQA approach from Section 2 to generate candidate plans and then label each augmented context $\tilde{x}$ (i.e., context combined with plans) with the correct label (here and in Section 3.3, we assume that there is a *unique* correct candidate plan; we provide an extension to multiple acceptable options in Section A2). We thus obtain a calibration set $Z=\{z_i=(\tilde{x}_i,y_i)\}_{i=1}^N$ with pairs of augmented contexts and correct labels.

Calibration. Next we follow Section 3.1 to perform calibration: first adjust $\hat{\epsilon}$ to achieve the $1-\epsilon$ coverage based on Eq. (2) and then find the quantile level $\hat{q}$. Given a new context $\tilde{x}_{\text{test}}$ (after MCQA in a new scenario) at test time, we can construct the calibration set $C(\tilde{x}_{\text{test}})$ that contains y_{test} with $1-\epsilon$ probability.

Triggering help. If $C(\tilde{x}_{\text{test}})$ is a singleton, the robot executes the corresponding plan. Otherwise, we deem the LLM uncertain over possible actions and trigger human help. The robot presents the human with $C(\tilde{x}_{\text{test}})$ (including the corresponding plans in text) and asks the human to choose one. The human

chooses y_{test} if $y_{\text{test}} \in C(\tilde{x}_{\text{test}})$[2], or halts the operation otherwise. This setup turns the coverage guarantee from CP to the task completion guarantee:

Proposition 1 (Single-step uncertainty alignment) *Consider a single-step setting where we use CP with coverage level $1-\epsilon$ to generate prediction sets and seek help whenever the set is not a singleton. With probability $1-\delta$ over the sampling of the calibration set, the task completion rate over new test scenarios drawn from $\mathcal{D}$ is at least $1-\epsilon$. If $\hat{f}$ models true conditional probabilities, the average prediction set size is minimized among possible prediction schemes that achieve $1-\epsilon$ completion rate.*

The proof immediately follows from the fact that under the assumption of accurate human help, the robot fails only when the prediction set does not contain the true label; the prediction set minimality follows Eq. (3). Thus, our approach addresses the goals of calibrated confidence and minimal help from Section 2.

3.3 Multi-Step Uncertainty Alignment

Now we extend the CP-based uncertainty alignment approach to settings where the LLM plans in multiple timesteps. This setting can be helpful when the LLM receives feedback from the environment or human between steps. However, the original CP formulation cannot be applied here since the context x^t between steps are dependent; moreover, the robot's actions at step t influence the distribution over contexts that the robot observes at future steps. Thus, the i.i.d. assumption for the coverage guarantee is no longer valid. Here, we present a novel extension of CP to multi-step settings that tackles this challenge.

Sequence-level calibration. The key ideas are to (i) *lift* the data to sequences, and (ii) perform calibration at the sequence level using a carefully designed nonconformity score function (that allows for causal construction of the prediction set at test time). Suppose that each data point consists of a sequence of augmented context $\overline{x} = (\tilde{x}^0, \tilde{x}^1, \ldots, \tilde{x}^{T-1})$ and true labels $\overline{y} = (y^0, y^1, \ldots, y^{T-1})$, where T is the time horizon and $\tilde{x}^t$ arises from having performed the *correct* actions in previous steps. The distribution $\mathcal{D}$ over scenarios induces a distribution over data sequences. We can again collect a calibration set $\overline{Z} = \{\overline{z}_i = (\overline{x}_i, \overline{y}_i)\}_{i=1}^N$. Next we use the lowest score over the timesteps as the score for the sequence[3]:

$$\hat{f}(\overline{x})_{\overline{y}} := \min_{t \in [T]} \hat{f}(x^t)_{y^t}. \tag{4}$$

With the standard calibration procedure in Section 3.1, we construct a *sequence-level* prediction set $\overline{C}(\overline{x}_{\text{test}}) := \{\overline{y} \in \mathcal{Y}^T \mid \hat{f}(\overline{x}_{\text{test}})_{\overline{y}} \geq 1-\hat{q}\}$ for a new context sequence $\overline{x}_{\text{test}}$ with the quantile level $\hat{q}$.

Causal construction of $C(\overline{x})$ at test time. Note that $\overline{C}(\overline{x}_{\text{test}})$ is constructed with the full sequence $\overline{x}_{\text{test}}$ at once. However, at test time, we do not see the entire sequence of contexts all at once but rather x^t_{test} one at a time. We thus need to construct $\overline{C}(\overline{x}_{\text{test}})$ in a *causal* manner (i.e., always relying only on current/past information). Consider the causally constructed prediction set $C^t(x^t_{\text{test}}) := \{y^t \mid \hat{f}(x^t_{\text{test}})_{y^t} \geq 1-\hat{q}\}$ at time t using the same quantile level $\hat{q}$ from the non-causal calibration above, and define $C(\overline{x}_{\text{test}}) := C^0(x^0_{\text{test}}) \times C^1(x^1_{\text{test}}) \times \cdots \times C^{T-1}(x^{T-1}_{\text{test}})$. We would like to obtain a lower bound on the sequence-level coverage, $\mathbb{P}(\overline{y}_{\text{test}} \in C(\overline{x}_{\text{test}})) \geq 1-\epsilon$.

Claim 1 *For any $\overline{y} \in \mathcal{Y}^T$, $\overline{y} \in \overline{C}(\overline{x}_{test}) \iff \overline{y} \in C(\overline{x}_{test})$.*

Proposition 2 (Multi-step uncertainty alignment) *Consider a multi-step setting where we use CP with coverage level $1-\epsilon$ to causally construct the prediction set and seek help whenever the set is not a singleton at each timestep. With probability $1-\delta$ over the sampling of the calibration set, the task completion rate over new test scenarios drawn from $\mathcal{D}$ is at least $1-\epsilon$. If $\hat{f}$ models true conditional probabilities, the average prediction set size is minimized among possible prediction schemes that achieve $1-\epsilon$ completion rate.*

The proofs are deferred to Section A1. Claim 1 allows us to construct causal prediction sets from non-causal calibration, and then we show the sequence-level task completion rate guarantee still holds.

Multiple acceptable options. Often, there can be multiple acceptable options at the same timestep, e.g., the task is to bring the human a soda, and either the Coke or Sprite on the table is acceptable. In such settings, we would like the prediction set to contain *at least one* acceptable option. We extend our method and confidence guarantees to this setting for both single- and multi-step problems in Section A2 and Section A3.

[2] If the correct option in $C(\tilde{x}_{\text{test}})$ is 'E', the human provides the correct action that was not listed by the robot. The robot continues with this action.

[3] We overload notation here and use $\hat{f}$ to also assign confidence scores to sequences.

4 Experiments

We evaluate our framework in a diverse set of language-instructed tasks and environments below, and demonstrate its effectiveness in achieving a user-specified task completion rate while minimizing user help. We use PaLM-2L [13] as the LLM in all examples unless otherwise noted. Additional discussions and details of experiment setup, robot perception, low-level policies are included in Section A8, Section A6, and Section A7.

Baselines. A straightforward way to construct prediction sets given a desired $1-\epsilon$ coverage is to rank options according to confidence and construct a set such that the cumulative confidence exceeds $1-\epsilon$; we consider two baselines that are based on such cumulative thresholding but use different kinds of scores: (1) **Simple Set** uses the same $\hat{f}$ as KNOWNO; (2) **Ensemble Set** [14] instead uses the frequencies of the LLM outputting $y \in \mathcal{Y}$ (out of 20 trials total) with randomized prompting. However, such ways of constructing prediction sets are not guaranteed to achieve $1-\epsilon$ coverage as the probabilities can be miscalibrated [15], and often include additional unnecessary options [16]. Furthermore, they do not provide any statistical guarantees. We also introduce a more modern method **Prompt Set** where the LLM directly outputs the prediction set (e.g., "Prediction set: [A, C]"). Lastly, we consider **Binary** where the LLM directly outputs a binary indicator for uncertainty (e.g., "Certain/Uncertain: Certain"); this setup is used in other LLM-based planning work [2] for triggering human intervention. Note that the ϵ level is not used in Prompt Set or Binary, and so the user cannot explicitly control the task success rate.

4.1 Simulated Tabletop Rearrangement

A robot arm is asked to rearrange objects on a table in the PyBullet simulator [17] (Fig. 1 right middle). Each scenario is initialized with three bowls and blocks of green, yellow, and blue colors. The task is to move a certain number of blocks or bowls towards a different object or at a specific location around it. We introduce three settings based on different types of ambiguities in the user instruction: (1) *Attribute* (e.g., referring to the bowl with the word "receptacle"), (2) *Numeric* (e.g., under-specifying the number of blocks to be moved by saying "a few blocks"), and (3) *Spatial* (e.g., "put the yellow block next to the green bowl", but the human has a preference over placing it at the front/back/left/right). For each setting, we construct a distribution over scenarios and perform experiments separately.

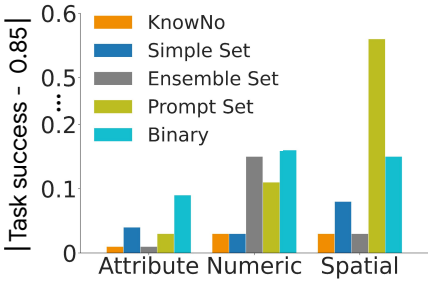

Figure 3: Error from specified task success level $1-\epsilon=0.85$ to the empirical success rate for the three settings in the simulated example.

KNOWNO achieves target task success rate consistently. First, we investigate whether KNOWNO and the baselines achieve a given target task success rate *consistently* in the three settings — we set the failure level $\epsilon = 0.15$. In Fig. 3 we show the difference between achieved and target rates for all methods. Results show that KNOWNO provides task success rates closest to the target overall, due to the coverage guarantee from CP. Simple Set and Ensemble Set cannot achieve coverage consistently. Prompt Set and Binary have a larger difference from the target since the user has no control over the error rate.

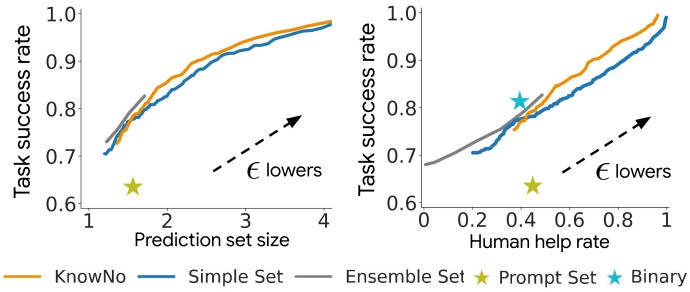

Figure 4: Comparison of task success rate vs average prediction set size (Left) and vs. human help rate (Right) in the simulated example averaged over the three settings. 200 trials are run for each method. ϵ is varied from 0.25 to 0.01 for KNOWNO, and from 0.6 to 0.01 for Simple Set and Ensemble Set.

KNOWNO achieves high task success rate with lower human help as ϵ varies. In Fig. 4 we vary the target error rate ϵ and show the full curves of task success rate vs. prediction set size and human help rate averaged over the three settings. For KNOWNO, Simple Set, and Ensemble Set, specifying a lower ϵ improves the empirical task success rate while also requiring more human intervention. The most natural comparison is between KNOWNO and Simple Set; both use next-token probabilities from the LLM as the confidence score but KNOWNO applies CP and Simple Set applies cumulative thresholding. KNOWNO achieves higher success-to-help ratios across ϵ levels, thanks to calibrated confidence from CP. Meanwhile,

Prompt Set and Binary do not allow controlling success rates. Prompt Set performs the worst, indicating the challenge of prompting-based methods for calibrated prediction set. Binary performs favorably at certain success levels but is (1) inflexible and (2) does not provide human with a prediction set for easier feedback.

Ensemble Set can perform well but is computationally expensive. Fig. 4 also shows that Ensemble Set provides high task success rate with small amount of human help at higher ϵ levels. However, there are two main drawbacks. First, we find in some scenarios that even with 20 randomized prompts, the LLM can fail to choose the correct option and thus assigns zero probability to it. As shown in Fig. 4, this means that Ensemble Set can fail to improve once it reaches some level of human help. Second, it requires $20\times$ inference time compared to other methods. Investigating how to lower the computational burden and combining ensemble-based probabilities with CP can be a fruitful future direction.

4.2 Real Multi-Step Tabletop Rearrangement

In this example, a real UR5 robot arm is asked to sort a variety of toy food items on a table (Fig. 5). In each scenario, three items are placed on the table initially, and the task is to sort them based on human preferences; we simulate a human with strong preferences for healthy food like eggs and fruits, and dislike for less healthy food like donuts and Skittles candies. To introduce ambiguities, the context for the LLM reveals only a subset of the preferences. Here we consider a multi-step setting with possibly multiple acceptable options per step — the LLM plans the new step conditioned on the previous action taken.

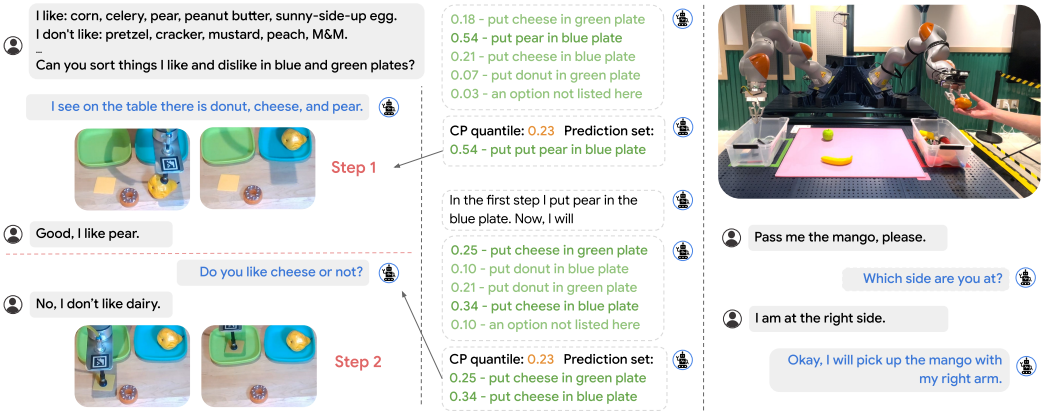

Figure 5: (Left) Multi-step CP is applied in the real tabletop rearrangement example. (Right) CP models ambiguity in possible human locations and triggers clarification from human in the bimanual setup.

KnowNo reduces step-wise and trial-wise intervention rate in multi-step setting. We compare KnowNo with Simple Set with the desired error level $\epsilon = 0.25$. Since Simple Set does not provide coverage, we first find $\epsilon = 0.42$ for Simple Set that achieves the same planing error rate as KnowNo in simulation. Then we run 50 trials for both methods with real hardware. Table 1 shows KnowNo reduces the human help rate by 14% step-wise and 8% trial-wise, while also reducing the average prediction set size. Compared to Simple Set that uses a much higher ϵ, KnowNo achieves *trial-level* task success rate precisely leveraging the Multi-Step Uncertainty Alignment from Sec. 3.3.

Method	$1-\epsilon$	Plan Succ	Task Succ	Set Size	Help-Step	Help-Trial
KnowNo	0.75	0.76	0.74	**1.72**	**0.58**	**0.92**
Simple Set	0.58	0.76	0.72	2.04	0.72	1.00

Table 1: Results for real tabletop rearrangement over steps. Plan success rate is fixed between the two methods for comparing the other metrics.

Bimanual manipulation. We additionally present results for a bimanual object rearrangement setup where ambiguities arise from the choice of the arm (Fig. 5 right); results are deferred to Section A4.

4.3 Real Mobile Manipulator in a Kitchen

In this example, each scenario involves a mobile manipulator in front of a countertop and next to a set of recycling/compost/landfill bins in a real office kitchen (Fig. 1). The tasks include picking up some object on the counter, and possibly putting it in the drawer, or disposing of it in one of the bins. For the distribution of possible scenarios, we introduce new types of ambiguities based on Winograd Schemas [18] (e.g., "There is the apple and bottled water on the counter...it is rotten. Can you dispose of it?"), and ones

that potentially involve unsafe actions (e.g., "place the bowl in the microwave."; there is a plastic bowl and a metal bowl, but only the plastic one is safe for the microwave). In Table 2, we compare KNOWNO to Simple Set again by first setting $\epsilon = 0.15$ and finding $\epsilon = 0.24$ for Simple Set that achives the same plan success rate. The real experiment results again show that KNOWNO reduces the human help rate by 14% and also reduces the average prediction set size.

Method	Model	$1-\epsilon$	Plan Succ	Task Succ	Set Size	Help
KNOWNO	PaLM-2L	0.85	0.87	0.76	2.22	**0.67**
Simple Set	PaLM-2L	0.76	0.87	0.75	2.38	0.81
KNOWNO	PaLM-2L-IF	0.85	0.86	-	**1.86**	**0.67**
KNOWNO	GPT-3.5	0.85	0.87	-	2.50	0.86

Table 2: Real mobile manipulator. Plan success rate is fixed for both methods to compare the other metrics.

Varying LLM choice. We also run KNOWNO with two other LLMs (without hardware evaluation). First, we use an instruction-finetuned version of PaLM-2L (PaLM-2L-IF); there is no significant performance difference from PaLM-2L but it generates smaller prediction sets in general by reducing the number of larger prediction sets (size 3 and 4) significantly. Second, we run GPT-3.5 (*text-davinci-003*) from OpenAI. However, we find it exhibits significant MCQA bias towards options D and E and against A and B, affecting the overall performance. We cannot run GPT-4 since it does not provide access to token probabilities.

5 Related Work

LLMs for robot planning and interaction. Large language models have shown to exhibit a wide range of capabilities: reasoning [19, 20], logic [21, 22], math [23], high-level planning [1, 24, 25, 26, 27, 28] with language feedback [29, 2], and writing robot code [5, 30, 31]. The generated outputs can be guided to a certain extent with sufficient prompting, but LLMs are still prone to confidently hallucinating outputs (e.g., referring to objects not observed in the scene [29], or calling motion primitive APIs that may not exist [5]). We hypothesize that these challenges can be alleviated while obtaining statistical guarantees by modeling the uncertainty of LLMs and aligning their confidence to a calibration set via conformal prediction.

Uncertainty quantification for LLMs. Motivated by LLMs' overconfidence and hallucination, there has been a growing body of work in quantifying and better calibrating uncertainty [32, 33, 34, 35, 36, 37, 38]. In contrast to typical calibration methods that associate uncertainty with point-valued outputs, CP-based methods for language modeling provide coverage guarantees for set-valued predictors [39, 40, 41, 42]. However, there has been no application of CP in quantifying uncertainty of LLMs with *free-form* outputs: Kumar et al. [43] apply CP to next-token prediction in MCQA tasks exclusively, while KNOWNO builds on MCQA but is applicable to general natural language generation tasks.

Conformal prediction in robotics. To the best of our knowledge, this work is the first to employ CP for language-based planning. Prior work has utilized CP for fault detection, trajectory prediction, and planning in dynamic environments [44, 45, 46, 47]. At each point in the planning horizon, the probabilistic safety guarantee either holds on average [47, 44], or is too conservative due to union bounding [45], or requires additional calibration data to reduce conservatism [46]. In contrast, we provide a novel multi-step extension to CP to guarantee correctness for the entire planning horizon by performing sequence-level calibration in settings where the robot's actions influence the distribution of future inputs.

6 Discussion

Summary: We propose KNOWNO, a framework that applies conformal prediction (CP) to address the problem of uncertainty alignment for language-instructed robots, which we formalize as providing statistical guarantees of task completion while minimizing human help. Experiments across a variety of simulated and real setups demonstrate significant improvements in performance and autonomy as compared to baseline approaches that lack formal assurances. **Limitations and future work:** The primary limitation of our work is that the task completion guarantee assumes environments (objects) are fully grounded in the text input to the LLM, and the actions proposed by the LLM planner can be executed successfully. In the future, we are looking to incorporate uncertainty of the perception module (e.g., vision-language model) and the low-level action policy (e.g., language-conditioned affordance prediction) into the CP calibration. Another exciting direction is to combine our methods with active preference learning [48, 49, 50, 51] to generate open-ended queries that maximally reduce uncertainty about human preferences. On the theoretical front, modifying CP to optimize different metrics for human help (e.g., minimizing human intervention rate by maximizing number of singleton sets) would be of practical interest. Overall, we hope that the work presented here spurs further efforts towards uncertainty alignment for safe and reliable language-instructed robots.

References

[1] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, et al. Do as I can, not as I say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[2] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, T. Jackson, N. Brown, L. Luu, S. Levine, K. Hausman, and B. Ichter. Inner monologue: Embodied reasoning through planning with language models. In *6th Annual Conference on Robot Learning*, 2022. URL https://openreview.net/forum?id=3R3Pz5i0tye.

[3] V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic Learning in a Random World*, volume 29. Springer, 2005.

[4] A. N. Angelopoulos, S. Bates, et al. Conformal prediction: A gentle introduction. *Foundations and Trends® in Machine Learning*, 16(4):494–591, 2023.

[5] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2022.

[6] K. Murray and D. Chiang. Correcting length bias in neural machine translation. *arXiv preprint arXiv:1808.10006*, 2018.

[7] A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shoeb, A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta, A. Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

[8] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[9] V. Vovk. Conditional validity of inductive conformal predictors. In *Asian Conference on Machine Learning*, pages 475–490. PMLR, 2012.

[10] M. Sadinle, J. Lei, and L. Wasserman. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234, 2019.

[11] Z. Jiang, J. Araki, H. Ding, and G. Neubig. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9: 962–977, 2021.

[12] Y. Bai, S. Mei, H. Wang, Y. Zhou, and C. Xiong. Efficient and differentiable conformal prediction with general function classes. *arXiv preprint arXiv:2202.11091*, 2022.

[13] Google, R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J. H. Clark, L. E. Shafey, Y. Huang, K. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G. H. Abrego, J. Ahn, J. Austin, P. Barham, J. Botha, J. Bradbury, S. Brahma, K. Brooks, M. Catasta, Y. Cheng, C. Cherry, C. A. Choquette-Choo, A. Chowdhery, C. Crepy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. Díaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. Garcia, S. Gehrmann, L. Gonzalez, G. Gur-Ari, S. Hand, H. Hashemi, L. Hou, J. Howland, A. Hu, J. Hui, J. Hurwitz, M. Isard, A. Ittycheriah, M. Jagielski, W. Jia, K. Kenealy, M. Krikun, S. Kudugunta, C. Lan, K. Lee, B. Lee, E. Li, M. Li, W. Li, Y. Li, J. Li, H. Lim, H. Lin, Z. Liu, F. Liu, M. Maggioni, A. Mahendru, J. Maynez, V. Misra, M. Moussalem, Z. Nado, J. Nham, E. Ni, A. Nystrom, A. Parrish, M. Pellat, M. Polacek, A. Polozov, R. Pope, S. Qiao, E. Reif, B. Richter, P. Riley, A. C. Ros, A. Roy, B. Saeta, R. Samuel, R. Shelby, A. Slone, D. Smilkov, D. R. So, D. Sohn, S. Tokumine, D. Valter, V. Vasudevan, K. Vodrahalli, X. Wang, P. Wang, Z. Wang, T. Wang, J. Wieting, Y. Wu, K. Xu, Y. Xu, L. Xue, P. Yin, J. Yu, Q. Zhang, S. Zheng, C. Zheng, W. Zhou, D. Zhou, S. Petrov, and Y. Wu. PaLM 2 technical report, 2023.

[14] Y. Lu, M. Bartolo, A. Moore, S. Riedel, and P. Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*, 2021.

[15] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

[16] A. Angelopoulos, S. Bates, J. Malik, and M. I. Jordan. Uncertainty sets for image classifiers using conformal prediction. *arXiv preprint arXiv:2009.14193*, 2020.

[17] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2022.

[18] H. Levesque, E. Davis, and L. Morgenstern. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2012.

[19] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[20] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

[21] M. Suzgun, N. Scales, N. Schärli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi, D. Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

[22] A. Creswell, M. Shanahan, and I. Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=3Pf3Wg6o-A4.

[23] A. Lewkowycz, A. Andreassen, D. Dohan, E. Dyer, H. Michalewski, V. Ramasesh, A. Slone, C. Anil, I. Schlag, T. Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022.

[24] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR, 2022.

[25] Y. Xie, C. Yu, T. Zhu, J. Bai, Z. Gong, and H. Soh. Translating natural language to planning goals with large-language models. *arXiv preprint arXiv:2302.05128*, 2023.

[26] Y. Ding, X. Zhang, C. Paxton, and S. Zhang. Task and motion planning with large language models for object rearrangement. *arXiv preprint arXiv:2303.06247*, 2023.

[27] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone. LLM+P: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

[28] J. Wu, R. Antonova, A. Kan, M. Lepert, A. Zeng, S. Song, J. Bohg, S. Rusinkiewicz, and T. Funkhouser. Tidybot: Personalized robot assistance with large language models. *arXiv preprint arXiv:2305.05658*, 2023.

[29] A. Zeng, M. Attarian, B. Ichter, K. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

[30] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

[31] E. Zelikman, Q. Huang, G. Poesia, N. D. Goodman, and N. Haber. Parsel: A (de-) compositional framework for algorithmic reasoning with language models. *arXiv preprint arXiv:2212.10561*, 2023.

[32] K. Zhou, D. Jurafsky, and T. Hashimoto. Navigating the grey area: Expressions of overconfidence and uncertainty in language models. *arXiv preprint arXiv:2302.13439*, 2023.

[33] Y. Xiao and W. Y. Wang. Quantifying uncertainties in natural language processing tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7322–7329, 2019.

[34] Y. Xiao, P. P. Liang, U. Bhatt, W. Neiswanger, R. Salakhutdinov, and L.-P. Morency. Uncertainty quantification with pre-trained language models: A large-scale empirical analysis. *arXiv preprint arXiv:2210.04714*, 2022.

[35] L. Kuhn, Y. Gal, and S. Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*, 2023.

[36] S. Kadavath, T. Conerly, A. Askell, T. Henighan, D. Drain, E. Perez, N. Schiefer, Z. Hatfield-Dodds, N. DasSarma, E. Tran-Johnson, S. Johnston, S. El-Showk, A. Jones, N. Elhage, T. Hume, A. Chen, Y. Bai, S. Bowman, S. Fort, D. Ganguli, D. Hernandez, J. Jacobson, J. Kernion, S. Kravec, L. Lovitt, K. Ndousse, C. Olsson, S. Ringer, D. Amodei, T. Brown, J. Clark, N. Joseph, B. Mann, S. McCandlish, C. Olah, and J. Kaplan. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

[37] S. Lin, J. Hilton, and O. Evans. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*, 2022.

[38] S. J. Mielke, A. Szlam, E. Dinan, and Y.-L. Boureau. Reducing conversational agents' overconfidence through linguistic calibration. *Transactions of the Association for Computational Linguistics*, 10:857–872, 2022.

[39] T. Schuster, A. Fisch, T. Jaakkola, and R. Barzilay. Consistent accelerated inference via confident adaptive transformers. *arXiv preprint arXiv:2104.08803*, 2021.

[40] T. Schuster, A. Fisch, J. Gupta, M. Dehghani, D. Bahri, V. Tran, Y. Tay, and D. Metzler. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022.

[41] N. Dey, J. Ding, J. Ferrell, C. Kapper, M. Lovig, E. Planchon, and J. P. Williams. Conformal prediction for text infilling and part-of-speech prediction. *The New England Journal of Statistics in Data Science*, 1(1):69–83, 2022. ISSN 2693-7166. doi:10.51387/22-NEJSDS8.

[42] P. Giovannotti and A. Gammerman. Transformer-based conformal predictors for paraphrase detection. In *Conformal and Probabilistic Prediction and Applications*, pages 243–265. PMLR, 2021.

[43] B. Kumar, C. Lu, G. Gupta, A. Palepu, D. Bellamy, R. Raskar, and A. Beam. Conformal prediction with large language models for multi-choice question answering. *arXiv preprint arXiv:2305.18404*, 2023.

[44] R. Luo, S. Zhao, J. Kuck, B. Ivanovic, S. Savarese, E. Schmerling, and M. Pavone. Sample-efficient safety assurances using conformal prediction. In *Algorithmic Foundations of Robotics XV*, pages 149–169, Cham, 2023. Springer International Publishing. ISBN 978-3-031-21090-7.

[45] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas. Safe planning in dynamic environments using conformal prediction. *arXiv preprint arXiv:2210.10254*, 2022.

[46] L. Lindemann, X. Qin, J. V. Deshmukh, and G. J. Pappas. Conformal prediction for STL runtime verification. In *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023)*, pages 142–153, 2023.

[47] A. Dixit, L. Lindemann, S. Wei, M. Cleaveland, G. J. Pappas, and J. W. Burdick. Adaptive conformal prediction for motion planning among dynamic agents. *arXiv preprint arXiv:2212.00278*, 2022.

[48] D. Sadigh, A. D. Dragan, S. S. Sastry, and S. A. Seshia. Active preference-based learning of reward functions. In *Proceedings of Robotics: Science and Systems (RSS)*, July 2017. doi:10.15607/RSS.2017.XIII.053.

[49] E. Biyik. *Learning Preferences For Interactive Autonomy*. PhD thesis, EE Department, Stanford University, 2022.

[50] D. S. Brown, W. Goo, and S. Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on Robot Learning*, pages 330–359. PMLR, 2020.

[51] V. Myers, E. Biyik, and D. Sadigh. Active reward learning from online preferences. In *International Conference on Robotics and Automation (ICRA)*, 2023.

[52] A. Kamath, M. Singh, Y. LeCun, G. Synnaeve, I. Misra, and N. Carion. MDETR-modulated detection for end-to-end multi-modal understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1780–1790, 2021.

[53] M. Minderer, A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen, et al. Simple open-vocabulary object detection with vision transformers. *arXiv preprint arXiv:2205.06230*, 2022.

[54] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[55] V. Agafonkin. Polylabel: a fast algorithm for finding the pole of inaccessibility of a polygon, July 2016. URL `https://github.com/mapbox/polylabel`.

[56] S. Singh, J.-J. Slotine, and V. Sindhwani. Optimizing trajectories with closed-loop dynamic SQP. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5249–5254. IEEE, 2022.

[57] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. RT-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.