Formal Dialogue Model for Language Grounding Error Recovery

Natasha Danas, Tim Nelson, Cobi Finkelstein, Shriram Krishnamurthi, Stefanie Tellex

Abstract—To enable humans to talk to robots, natural language commands need to be grounded into a symbolic goal, such as linear temporal logic, which the robot can then execute. However, natural language is often ambiguous and commands are often nuanced, making language grounding errors and robot mistakes inevitable. We address this problem by enabling the robot to ask questions that differentiate between the beam searched set of $k$-most-likely groundings. Maximal semantic differencing, a $k$-way extension to standard 2-way semantic differencing, allows the robot to ask clarifying questions about the groundings via differentiating trajectories, instead of asking about symbolic goals the user is not trained to interpret. The user can then clarify which trajectory satisfies their command, in turn clarifying the correct grounding. We evaluate the beam search, maximal semantic differencing, and user clarification components separately—then extrapolate to estimate the performance and accuracy of this dialog model as an end-to-end system in practice. With 1-2 seconds for high-level trajectories (3-20 seconds for 8x8 low-level grid-world trajectories), we expect the robot to recover from about 80% to 94-99% accuracy for unseen natural language commands, depending on user clarification accuracy.

I. INTRODUCTION

To enable humans to talk to robots, natural language (NL) commands must be grounded into a symbolic goal. However, NL is often ambiguous and commands are often nuanced, making language grounding errors and the robot mistakes inevitable. The current state of the art grounds these commands into linear temporal logic (LTL) \[7, 15\]. A Markov Decision Process (MDP) interprets the goal into a trajectory: concrete examples of the LTL goal being satisfied (positive) or not satisfied (negative) through particular robot behavior. The commands are grounded via a neural network sequence-to-sequence (seq2seq) model with attention that is trained to handle arbitrary NL commands using data from a crowd-sourced corpus.

The latest language grounding model \[15\] only achieves about 80% accuracy on the unseen commands, with no recovery technique for the remaining 20%. Previous dialog models have developed question-answer protocols to handle ambiguity of environment observations and execution failures \[8, 16\]. However, the generated questions are either a closed finite set, or promise limited information gain.

We present the three individual components of a dialogue model for the robot to recover from language grounding errors by asking targeted clarifying questions, as shown in Fig. 1. First, we implement beam search within the seq2seq model to produce the $k$-most-likely LTL groundings for a user provided NL command, for an arbitrary beam width $k$. Second, we define and implement maximal semantic differencing to identify differentiating trajectories between the $k$-most-likely LTL formulae, to avoid asking the user about symbols they are not trained to interpret. Third, we study crowd-sourced users’ ability to clarify whether a trajectory satisfies a provided NL command. This clarification can be used to recover the LTL formula that represents the trajectory and also satisfies the original NL command.

We define maximal semantic differencing as change impact analysis between any number of specifications, as opposed to the standard semantic differencing between two specifications. For example, semantic differencing can be used to look at the change impact between two firewall policies: what packets are dropped by one and accepted by the other, and vice-versa \[13\]. For our use case, the specification is not a firewall policy, but the robot’s environment, possible behavior, and LTL goals; the change impact being the differentiating trajectories that satisfy one LTL goal but not the others. We implement maximal semantic differencing as a modification to a formal methods tool already capable of performing standard semantic differencing.

We measure the effectiveness of beam search, maximal semantic differencing, and user clarification components separately by answering the following research questions:

1) How does increasing the number of most-likely groundings increase accuracy?
2) How do the number of most-likely groundings and abstraction-level of the environment affect the time to perform maximal semantic differencing?
3) How quickly and accurately can users clarify whether a trajectory satisfies a NL command?
We evaluate grounding variant accuracy by computing the position of the correct LTL formula in the $k$-most-likely groundings, for each NL command. The data is evaluated using 5-fold cross-validation. Maximal semantic differencing performance is evaluated over hand written specifications of varying sized low-level grid-world environments; we also evaluate a high-level environment with grid-points abstracted out, as that level of detail is not necessary to produce differentiating trajectories. We assess user clarification ability by performing another crowd-sourced study on the NL commands workers had trouble generating for the seq2seq training corpus, except they now only need to discriminate between the given command and trajectory. We extrapolate and estimate that for a reasonably sized environment, with 1-2 seconds for high-level trajectories (3-20 seconds for 8x8 low-level grid-world trajectories), we expect the robot to recover from about 80% to 94-99% accuracy for unseen natural language commands, depending on user clarification accuracy.

II. RELATED WORK

Deits et al. [5] present an information-theoretic approach for clarifying ambiguities in the NL command, while Tellex et al. [16] present a graph-theoretic approach for communicating environment induced robot execution failures. Both of these approaches are complementary to our model-theoretic approach, which clarifies ambiguities in the LTL groundings.

Lignos et al. [11] define a similar formal dialogue to our approach, allowing a user to control a robot through a search-and-rescue environment via NL commands. However, in the case of a grounding error, their dialogue only points out the parts of the NL command that need to be restated. Instead, we have them discriminate between trajectories to avoid asking the user to continually restate their intent.

Boteanu et al. [1] present a grounding model that can synthesize full robot controllers, and verify that the resulting controllers assumptions about the environment and the interpretation of the user’s goals. However, the verification stage cannot determine whether the instructions progress towards the users intended goal. Our approach will improve accuracy of interpreting the users intentions, putting us one step closer to bridging this gap.

Boteanu et al. [2] use a formal methods approach to perform goal repair for unsatisfiable scenarios using hard-coded NL interactions to weaken contradictory assumptions about the environment. In most cases, even incorrectly grounded goals are often still satisfiable, and possibly equivalent to a correct grounding, for a given environment. Our approach covers this ignored majority of semantic error cases.

Whitney et al. [18] propose a FETCH-POMDP in which the robot enumerates through the items they believe the user wants them to retrieve, and asks whether each was the intended item. Both the knowledge-base and question-answer protocol are hard-coded, and cannot ask questions beyond the objects they are specified for. Our approach can handle an arbitrary set of LTL expressions for a given environment.

III. TECHNICAL APPROACH

We beam search the $k$-most-likely LTL groundings for a given NL command, then perform maximal semantic differencing to produce a set of differentiating trajectories, and finally ask the user which instance is correct.

A. Grounding Variant Generation

We implement beam search within the seq2seq model to produce the top-$k$ likely LTL groundings for a given NL command, for an arbitrary beam width $k$. Beam search is a well known algorithm with long standing history in machine translation systems[12]. While it is an incomplete search, the additional cost of performing a breadth-first search with a log-likelihood based branching heuristic is negligible, especially for small beam widths. As reported in our evaluation, a beam width of $k = 10$ suffices for effectively 100% accuracy recovery, thus we did not investigate more sophisticated or alternative search techniques.

B. Standard Semantic Differencing

Before the robot can identify differentiating trajectories for the top-$k$ candidate groundings that it can ask the user about, we first introduce standard semantic differencing, which can produce these trajectories for $k = 2$. Semantic differencing can be specified within a model finder, a class of formal methods tools which take a formal specification as input and output models: concrete examples that satisfy the specified set of logical constraints. The robot’s environment, behavior, and goals are an input specification that is satisfied by a certain set of concrete trajectories of the robot through the state/action space. Note, the stochastic or partially-observable features are left out of the environment specification, as these tools are useful for knowledge bases and high-level planning—not for low-level planning which is left to MDP-based approaches.

We compute semantic differences using the model finder Alloy [8]. Alloy turns each first-order relational logic constraint into propositional logic, via its compiler named Kodkod [17]. The resulting propositional logic formulae are then passed to a SAT solver [14], to find a satisfying assignment: the model or trajectory. For example[4] let us consider two possible groundings for avoid room 1 until you go to landmark 1: the correct grounding ($\neg room_1$ U landmark_1) and the incorrect grounding ($\neg room_1$ U landmark_1 U room_1).

We can semantically differentiate these goals by running Alloy, to produce trajectories that satisfy both (describes commonality), and just one of each goal (describes independence). In this case, the first run (A and B) is satisfiable, the second run (A and not B) is unsatisfiable, and the third run (B and not A) is satisfiable. By the results of these three runs, we can conclude that notRoom1UntilLandmark1 implies notLandmark1UntilRoom1 for this specific environment: due to the fact Landmark1 is located in Room1. For other pairs of LTL goals or other environments, we may instead conclude that some goals are unsatisfiable (have no satisfying

1The specification has been omitted for space, but can be viewed [here](#).
trajectories), are equivalent (same set of trajectories), are partially disjoint (some common and independent trajectories), or some goals are completely disjoint (only independent trajectories).

C. Maximal Semantic Differencing

Given the top-\(k\) candidate groundings, the robot needs to identify differentiating trajectories that it can ask the user about. To compute these trajectories, we specify maximal semantic differencing as change impact analysis between any number of specifications, as opposed to just two. Our implementation enables us to turn a set of LTL formulae into a set of trajectories that describe each formula most independently from the others. We do this by extending Alloy to support soft constraints, which can be optionally satisfied unlike the usual hard constraints.

```
// satisfy - ( red_room \lor \neg green_room ), minimize satisfaction of others
run { Robot.whereFirst[1] = gh4w
notRedRoomUntilGreenRoom
soft not eventually\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\neg\ne
many LTL operands and parentheses. We do not expect our technique to work well in these cases, but a more qualitative evaluation with principled choices in training data is required to fully report on the generalization problems that remain. Of course, combinatory categorical grammar (CCG) based approaches will generalize better than our seq2seq approach without requiring user intervention.

**B. Maximal Semantic Differencing Performance**

We break down maximal semantic differencing performance into three metrics: translation time, UNSAT solve time, and SAT solve time. We pay a cost of one translation time, plus one solve time per variant, depending on whether the query is satisfiable or not.

<table>
<thead>
<tr>
<th>Gridworld Size</th>
<th>Trajectory Length (time size)</th>
<th>Number of Variants (queries)</th>
<th>Translation time (average over all queries)</th>
<th>UNSAT Solve time (average over UNSAT queries)</th>
<th>SAT Solve time (average over SAT queries)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6x6 (high-level)</td>
<td>~5</td>
<td>6</td>
<td>90 ms</td>
<td>5 ms</td>
<td>65ms</td>
</tr>
<tr>
<td>4x4 (low-level)</td>
<td>~5</td>
<td>2</td>
<td>166 ms</td>
<td>202 ms</td>
<td>528 ms</td>
</tr>
<tr>
<td>8x8 (low-level)</td>
<td>~10</td>
<td>6</td>
<td>1386 ms</td>
<td>299 ms</td>
<td>1774 ms</td>
</tr>
<tr>
<td>16x16 (low-level)</td>
<td>~20</td>
<td>12</td>
<td>11 sec</td>
<td>21 sec</td>
<td>128 sec</td>
</tr>
</tbody>
</table>

Since each query is either UNSAT or SAT, we can view UNSAT solve time as a lower bound, and SAT solve time as an upper bound. So, in order to process 10 groundings for an 8x8 grid-world, we have to wait 3-20 seconds to produce every maximally independent low-level trajectory. However, we only have to wait 1-2 seconds with the grid-points abstracted out, as that detail is not necessary to produce differentiating trajectories, and exponentially increases the size of the search space. Additionally, an average of 1-2 groundings are usually not grammatically correct, and can be thrown out before performing maximal semantic differencing. Also, many of the groundings end up being unsatisfiable, or equivalent to other groundings, due to the context of the environment. In practice, we expect performance to end up closer to the lower bound than the upper bound.

**C. User Ability to Clarify Differentiating Trajectories**

We evaluate user feasibility by their accuracy in discriminating whether a particular trajectory satisfies a given NL command. We report the distribution of their average scores on the 38 discrimination tasks given, for a sample size of \( n = 100 \).

**V. Conclusion**

We expand our view from one NL-LTL grounding to the \( k \)-most-likely groundings via beam search, perform maximal semantic differencing in a modified model finding process within Alloy that allows the robot to ask clarifying questions about the groundings via differentiating trajectories instead of logical forms. With 1-2 seconds for high-level trajectories (3-20 seconds for 8x8 low-level grid-world trajectories), the robot can now repair itself in almost all failure cases for unseen commands. At this point, Alloy is the only weakness in our approach, and there are many avenues of improvement. Enabling incremental solving of multiple semantic differencing queries will reduce average query time by an order of magnitude, as the SAT solver will not need to re-load and re-solve much of the search problem shared between queries. Taking a compositional approach [9] to map a differentiating high-level trajectory to some low-level trajectory should be faster than generating a differentiating low-level trajectory directly. We may also be able to implement low-level maximal semantic differencing directly into an MDP planner. Once we overcome the current weaknesses of the individual components, future work will perform a full end-to-end evaluation and demonstration on a real robot.
RESEARCH ARTIFACTS

- Sequence-to-Sequence Grounding Variant Modifications
- Alloy Soft Clause Modifications
- Maximal Semantic Differencing Specifications
- Differentiating Trajectory Clarification User Study

REFERENCES


