Benchmarking Clarifying Questions for Effective Collaboration in Grounded Instruction-Based Interactions

Anonymous ACL submission

Abstract

Motivated by the adaptability of human intelligence across various tasks and multi-modal environments, the research community is actively engaged in developing interactive agents capable of engaging in natural conversations with humans and assisting them in real-world tasks. These agents need the ability to request feedback in the form of situated clarifying questions when communication breaks down or instructions are unclear. This paper delves into an extensive investigation of the production of clarifying questions within the context of human-centered AI instruction-based interaction, using a Minecraft environment as a grounding framework. The unique challenges presented by this scenario include the agent's requirement to navigate and complete tasks in a complex, virtual environment, relying on natural language instructions and action states.

In this paper, we made the following contributions: (i) a crowd-sourcing tool for collecting grounded language instructions along with clarifying questions in times when instructions are not clear at scale with low costs; (ii) a substantial dataset of grounded language instructions accompanied by clarifying questions; and (iii) several state-of-the-art baselines for requesting feedback in case of unclear instructions. These contributions are suitable as a foundation for further research.

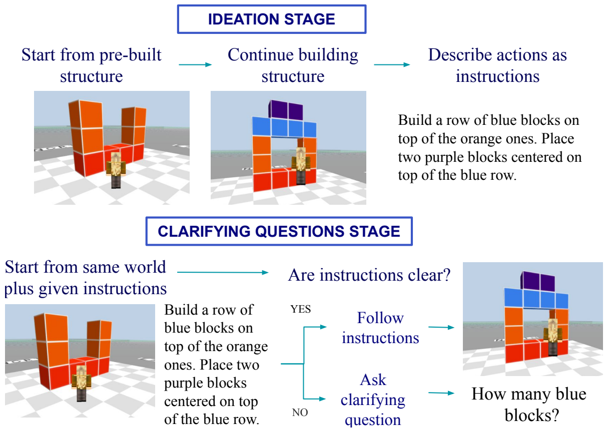


Figure 1: An example of human-agent interactive collaboration, where the goal is to build a given structure, and the agent needs to decide whether to follow the instruction or ask a clarifying question

1 Introduction

One of the long-lasting goals of AI agents (Winograd, 1972) is the ability to seamlessly interact with humans in natural language to help humans learn new skills (Narayan-Chen et al., 2019a; Kiseleva et al., 2022a; Zhang et al., 2021; Wang et al., 2023a) or assist in solving tasks (Shridhar et al., 2019; Kiseleva et al., 2022b). To achieve the latter, the agent must understand and respond to human language to execute instructions in a given environment (Skrynnik et al., 2022; Kiseleva et al., 2022a,b). Over the years, researchers have proposed many tasks to tackle this human-AI collaboration challenge, many centered around humans providing instructions to the agent to solve a goal (Gluck and Laird, 2018; Shridhar et al., 2020). An example is the blocks world task, where the agent must understand human instructions to move blocks on a grid (Winograd, 1972; Bisk et al., 2016). Other setups use Minecraft (Gray et al., 2019a), to move objects around (Abramson et al., 2020), to simulate human behavior (Park et al., 2023), or to simulate household tasks (Shridhar et al., 2019; Wang et al., 2023b). However, the instructions humans provide are often inherently ambiguous for most tasks. To complete these tasks successfully, agents need to engage in conversation by asking clarifying questions (Aliannejadi et al., 2021a; Shi et al., 2022; Press et al., 2022), which creates a naturally friendly interface for humans (Nass and Moon, 2000).

We aim to provide an in-depth investigation into the production of clarifying questions for grounded instruction-based interaction using a Minecraft environment, which has shown its effectiveness in studying human-AI collaboration (Fan et al.; Wang et al., 2023a; Kanervisto et al., 2022). This scenario presents a unique challenge, as the agent must

Table 1: Comparison between relevant platforms.

Dataset	Settings	Size of dataset	Collaborative instructional (AI/Human)	Availability of Data collection tool	Availability of Training environment
SHRDLURN(Wang et al., 2016)	Building game	100 games (10,223 utterances)	✓	✗	✓
Voxelurn(Wang et al., 2017)	building structures	230 structures (36,589 utterances)	✓	✗	✓
CEREAL-BAR(Suhr et al., 2019)	collaborative game	1202	✓	✗	✓
ALFRED(Shridhar et al., 2019)	Household tasks	25,743	✗	✗	✓
CVDN(Thomason et al., 2019)	Navigation	2050	✓	✓	✓
TEACh(Padmakumar et al., 2022)	Household tasks	3215	✓	✗	✓
MineDojo (Fan et al., 2022)	Minecraft	730K YouTube videos, 7K Wiki pages, 340K Reddit posts	✓	N/A	N/A
MineRL (Guss et al., 2019)	Minecraft	500 video hours	✗	✗	✓
HoloAssist (Wang et al., 2023b)	Physical tasks	166 video hours	✓	N/A	N/A
Ours	**Collaborative** building	9,111 utterances/1,1142 clarifying questions	✓	✓	✓

navigate and complete tasks in a complex, virtual environment, relying solely on natural language instructions. To ensure successful task completion, the agent must identify gaps in the instructions and pose relevant clarifying questions, as demonstrated in Fig. 1. By tackling this problem head-on, we intend to pave the way for more effective and user-friendly human-AI agent interactions.

A significant challenge hindering the exploration of building interactive agents (Narayan-Chen et al., 2019b; Bara et al., 2021) is the scarcity of appropriate datasets, and scalable and easily extendable data collection tools. These deficiencies have impeded progress in the field and pose a considerable obstacle to developing effective solutions. Our work addresses this challenge by proposing a novel dataset and scalable data collection methodology, thus contributing to the field's progress. We believe our work will enable researchers to explore new avenues and enhance user experience in human-AI interactions by addressing this important obstacle. In summary, our main contributions are:

- **C1 Crowdsourcing Tool for Collecting Interactive Grounded Language Instructions** specifically designed for efficiently gathering interactive grounded language instructions within a Minecraft-like environment. With low costs, we can do so at a large scale because it does not require multiple players to be online simultaneously (Sec. 3).
- **C2 Extendable Dataset of Human-to-Human Grounded Language Instructions** that is accompanied by clarifying questions (Sec. 4). This dataset represents a valuable resource for various research directions, including but not limited to building structures based on given instructions or predicting clarifying questions.
- **C3 Baselines for Predicting Clarifying Questions** on the aforementioned dataset which serves as a benchmark for evaluating the performance of future models (Sec. 5).

2 Related Work

Natural Language Interfaces (NLIs) have been a subject of study in various disciplines, including human-computer interaction and information search, for several decades. Early works such as (Woods et al., 1972; Codd, 1974; Hendrix et al., 1978) laid the foundation for understanding and designing effective interfaces for human language communication with computers.

Evolution of NLIs and Applications: Tab. 1 demonstrates a comprehensive set of related platforms. In recent years, there has been a resurgence of interest in NLIs due to advances in language understanding capabilities driven by large-scale deep learning models (Devlin et al., 2018; Liu et al., 2019; Clark et al., 2020; Adiwardana et al., 2020; Roller et al., 2020; Brown et al., 2020; OpenAI, 2023; Chowdhery et al., 2022) and the increasing demand for various applications such as virtual assistants, dialog systems (Li et al., 2019, 2020c; Burtsev et al., 2017; Li et al., 2020b, 2021), and question answering systems (Liu and Lane, 2017, 2018; Dinan et al., 2020; Zhang et al., 2019). NLIs now extend beyond traditional databases to encompass knowledge bases (Copestake and Jones, 1990; Berant et al., 2013) to robots (Tellex et al., 2011), personal assistants (Kiseleva et al., 2016b,a), and other forms of interaction (Fast et al., 2018; Desai et al., 2016; Young et al., 2013; Su et al., 2017).

Agent Interactivity and Learning: The focus has shifted towards interactivity and continuous learning, enabling agents to interact with users, learning new tasks from instructions (Li et al., 2020a), assessing their uncertainty (Yao et al., 2019), asking clarifying questions (Aliannejadi et al., 2020a, 2021b; Arabzadeh et al., 2022), and leveraging feedback from humans to correct mistakes (Elgohary et al., 2020; Nguyen et al., 2022; Nguyen and au2, 2019). Currently, LLMs are also being studied to asses uncertainty and their own errors (Press et al., 2022; Ren et al., 2023).

Grounded Language Understanding: This paper focuses on grounded language understanding—connecting natural language instructions with real-world or simulated environment context and taking corresponding actions (Hermann et al., 2017; Mitsuda et al., 2022). This is crucial to enabling more effective communication between humans and intelligent agents. Our work focuses specifically on tackling grounded language understanding in the context of collaborative building tasks performed by agents, as highlighted in (Carta et al., 2023; Kiseleva et al., 2021, 2022b; Mehta et al., 2023; Mohanty et al., 2022; Skrynnik et al., 2022).

Leveraging Minecraft for Grounded Language Understanding: We select Minecraft for grounded language understanding due to its distinct advantages. Szlam et al. (2019) highlights the benefits of an open interactive assistant in Minecraft, offering a cost-effective alternative to real-world assistants. The game's 3D voxel gridworld and adherence to simple physics rules provide ample research scenarios for reinforcement learning experimentation. Minecraft's interactive nature, player interactions, and dialog exchanges offer diverse opportunities for grounded natural language understanding (Yao et al., 2020; Srinet et al., 2020; Narayan-Chen et al., 2019b). The game's immense popularity ensures enthusiastic player interaction, facilitating rich human-in-the-loop studies. Minecraft's advantage extends to the availability of the highly developed set of tools for logging agents interactions and deploying agents for evaluation with human-in-the-loop, including *Malmo* (Johnson et al., 2016), *Craftassist* (Gray et al., 2019b), *TaskWorldMod* (Ogawa et al., 2020), *MC-Saar-Instruct* (Köhn et al., 2020) and *IGLU GridWorld* (Zholus et al., 2022).

3 Data Collection Tool

Narayan-Chen et al., 2019b proposed a setup for *a collaborative building task* within the Minecraft environment where an Architect is provided with a target structure that needs to be built by the Builder. The Architect provides instructions through a chat on how to create the target structure, and the Builder can ask clarifying questions if an instruction is unclear (Zhang et al., 2021). This approach required installing Microsoft's Project Malmo (Johnson et al., 2016) client, which provides an API for Minecraft agents to chat, build, and the ability to save and load game states, which makes it limited to lab-based studies. The setup collects multi-turn interactions between the Architect and the Builder, collaboratively working towards building a given target structure. However, having multiple players online adds unnecessary complications, such as waiting while one of the players is typing, and costs.

We have developed and released an open-source data collection tool[1] that is specifically designed to facilitate the collection of data for multi-modal collaborative building tasks. Our tool eliminates the need for participants to install a local client and allows multiple participants simultaneously annotating data, consequently streamlining the data collection process. As such, it enables *1) Integration with Crowdsourcing platforms:* Our work has the ability to merge and integrate seamlessly into any crowdsourcing platforms for efficient participant scaling and collecting more data. *2) Bidirectional Dataset:* While most datasets are one-way, our dataset is bidirectional. It can be used to teach both architects and builders, facilitating more comprehensive language understanding in collaborative building tasks. and *3) Game Environment for Testing:* We employ a game-type environment, which is more scalable and easier for testing compared to video-based approaches. This choice of environment enhances the practicality and efficiency of our approach. We have used the Amazon Mechanical Turk (MTurk) as the crowd-sourcing platform after obtaining approval from the Institutional Review Board (IRB). Each participant or annotator submits a HIT (Human Intelligence Task). A HIT is comprised the CraftAssist (Gray et al., 2019b) voxelworld and a form which is customizable for different tasks. The form includes rules for a given task and a segment where task instructions or clarifying questions for the building task. The CraftAssist voxelworld is a framework that provides tools and a platform for dialog-enabled interactive agents that learn from natural language interactions. The library provides a 3-d voxelworld grid where agents perform building actions that can be recorded as action states and retrieved for following sessions. Current actions supported by the integrated CraftAssist framework include picking, placing, and removing blocks of different colors within the voxelworld. Agents can also jump to place blocks. These actions enable agents to create structures of varying complexity. Fig. 5 in the appendix illustrates the MTurk views of the task with the embedded voxelworld.

[1]`https://bit.ly/42ZUNf7`

Table 2: Statistics of Multi-Turn Dataset

Target Structures	31
Completed Games	127
Median Duration of Completed Games	16 mins
Utterances	811
Avg. Length of Instructions	19.32 words
Clarifying Questions	126

4 Datasets

We built corpora of multi-modal data, which could be used towards solving wide-ranging NLP and RL tasks, including training interactive agents by demonstrations given natural language instructions (Skrynnik et al., 2022). Our research initially concentrates on multi-turn interactions, following a similar approach as presented by (Narayan-Chen et al., 2019b) (Sec. 4.1). To enhance the size of our dataset, we subsequently expanded our data collection efforts to a Single-Turn dataset (Sec. 4.2) to gather a larger corpus of data more efficiently. The datasets and accompanying code for analysis and visualization is openly available [2] .

4.1 Multi-Turn Dataset

The Multi-Turn dataset comprises dialog-behavior sequences, which we called *game* (Appendix Fig. 2). The sequences either start from scratch for a given goal structure or build on intermediate results. In each *turn*, an annotator takes on the role of either the Architect or the Builder. Architects provide the next step instruction, while the Builder starts with an empty world and executes the instruction or poses a clarifying question. We have improved the data collection process by introducing asynchronous turn-taking. This means the tool no longer relies on having the same annotators online throughout the game. We have implemented checks to prevent a single annotator from taking on both architect and builder roles for the same structure. Importantly, this asynchronous approach allows for the simultaneous launch of multiple structures. Annotators can work on different structures concurrently without waiting for responses, saving time and making process scalable.

Tab. 2 shows the summary of the Multi-Turn dataset. There are 31 goal structures presented to annotators to build. We process and clean the data by filtering out missing and low-quality submissions such as very short instructions. Finally, we have 127 completed game sessions, with the median duration of a game being around 16 minutes. A game session is considered complete when the Builder can completes building a given goal structure after interacting with and following instructions provided by the Architect. This is denoted by the Architect marking the structure as *"complete"*. Across all the games, we had 811 utterances or dialog interactions between the Architect and Builder annotators. The average length of instructions provided by the Architects was around 19 words, and the number of clarifying questions asked by the Builders – 126 (for all the filtered games).

To provide a deeper understanding of the covered structures in our multi-turn dataset, we performed manual labeling on the 31 structures. The labels and the corresponding number of structures in the dataset in brackets, are as follows: 1. *flat [7]:* all blocks on the ground 2. *flying [27]:* there are blocks that cannot be fully-added without removing some other blocks 3. *diagonal [6]:* some blocks are adjacent (in the vertical axis) diagonally 4. *tricky [6]:* some blocks are hidden or they should be placed in a specific order 5. *tall [25]:* a structure cannot be built without the agent being high enough (the placement radius is 3 blocks) We consider different categories of the structures to make sure the agent is using different skills and abilities and also to make sure the target structures are diverse. For instance, if all the structures are flat, the agent will never learn to use other actions, such as flying. This diversity is essential for training a robust and adaptable agent.

4.2 Single-Turn Dataset

From our extensive study on Multi-Turn data collection, we identified certain challenges that crowd-sourced annotators encountered when engaging in the collaborative building task and issuing instructions for specific target structures. To enhance the crowd-sourcing process, we decided to simplify the task. Our approach involved removing the added complexity of building a predefined target structure. Instead, participants were free to perform free-form building actions within the voxelworld while providing instructions that should allow another worker to rebuild the same structure. This modification led to creating Single-Turn task segments, where annotators collaborated asynchronously to construct the same structure. With this adjustment, we were able to collect data at a faster pace, resulting in a larger corpus comprising of natural language instructions, corresponding actions performed based on those

[2] https://bit.ly/43WhnGC

instructions, and a set of clarifying questions. We record and save actions performed by annotators in a key-value pair format that stores the movement of the agent and positional changes of blocks within the voxelworld.

To provide diverse starting canvases for annotators, we utilized the Multi-Turn dataset to load different world states, which served as varying initial conditions for the building process. The process of collecting single-turn instructions and associated clarifying questions is in (Fig. 1):

- An annotator is assigned a world state from the Multi-Turn dataset as the starting point for their building task (Fig. 1: Ideation Stage).
- The annotator is prompted to perform a sequence of actions for a duration of one minute.
- Then, the annotator is required to describe their set of actions in the form of an instruction.
- Another annotator is shown the instruction and asked to perform the steps mentioned. If the instruction is unclear, the annotator specifies it as thus and asks clarification questions (Fig. 1: Clarification Question Stage).

The instructors answered these clarifying questions, and the data related to these clarifying questions has also been released with this dataset. Tab. 3 presents comprehensive statistics on the Single-Turn dataset, currently the largest dataset available for interactive grounded language understanding. We processed and cleaned the collected Single-Turn dataset by following a heuristic approach, which included filtering out samples where the length of instruction was very short. We also checked whether the instruction was in English and evaluated jobs to remove submissions by annotators who provided low-quality instructions, such as providing the same instruction repeatedly. As shown in Tab. 3, the Single-Turn corpus comprises 8,136 pairs of actions and instructions. On average, an instruction has 18 words, which indicates the instructions are descriptive enough for a one-minute of building.

In addition to the processing steps for cleaning instructions, for the clarifying questions, if an annotator marked the instruction as ambiguous, they were supposed to issue a clarifying question else the submission would be filtered out with a warning provided to the annotator. This was to ensure that every instruction annotated as "not clear" is accompanied by at least one clarifying question. Out of 8,136 instructions, 1,056 (12.98%) were annotated as *Not Clear*, thus being ambiguous, and 7,080 (87.02%) as *Clear* instructions. The average length of clarifying questions is around 12 words. Tab. 6 in the appendix exemplifies a few instructions marked as being unclear, along with clarifying questions issued by annotators. Majority of clarifying questions fall into the following categories:

- *Color*: Questions clarifying the color of the blocks to be used.
- *Direction/Orientation*: Questions clarifying the direction and orientation in the world.
- *Number of blocks*: Questions that clarify the number of blocks to be placed.
- *Identifying blocks to be changed*: Questions clarifying which blocks need to be changed.

It is important to note that we reassessed the annotations for 100 randomly selected instructions to gauge the level of agreement among the annotators. The agreement rate among the three annotators for these 100 instructions falls within the range interpreted as "fair" according to the Krippendorff agreement measure. This suggests that the interpretation of ambiguous instructions can be highly subjective, and moreover, emphasizes the complexity of such a task. While one annotator may perceive an instruction as clear, another may find it ambiguous. Furthermore, different annotators may ask different clarifying questions about the same instruction, as they may identify unclear aspects from various perspectives.

The Single-Turn approach offers several advantages over the sequential nature of the Multi-Turn process, one of which is the independence of each sample, allowing for easier utilization in different tasks. Each turn can be interpreted as a complete set of information, enabling flexibility and versatility in its application as they do not rely on the context of previous turns. This independence allows researchers to extract valuable insights and information from individual turns without considering the entire dialogue sequence. Furthermore, the Single-Turn approach allows for collecting multiple clarifying questions for each instruction augmenting the richness and diversity of the dataset, enabling a deeper understanding of the nuances and challenges in generating clarifying questions.

Table 3: Statistics of Single-Turn Dataset.

Instructions (train/test)		Avg. Length (in words)	
Total	8136 (6843/1293)	Instructions	18.29
Clear	7080 (5951/1129)	Clarifying Questions	12.05
Ambiguous	1056 (892/164)		

5 Baselines Models and Evaluation

We have developed baselines for the prediction of clarifying questions in the Architect-Builder task mentioned in (Sec. 4.2) using the Single-Turn dataset. As such, we focus on the following key research questions:

- **When to ask clarifying questions?**: Predicting whether an instruction provided by the Architect is ambiguous or insufficient for the Builder to complete a task successfully indicating further clarification is required.
- **What clarifying question to ask?** When faced with an instruction that is considered ambiguous, this research question focuses on determining the appropriate question to ask for clarification.

It is worth noting that issues related to determining *When* and *What* clarifying questions to ask have gained significant attention in the domains of NLP and information retrieval (IR) (Aliannejadi et al., 2019, 2021b, 2020b; Arabzadeh et al., 2022). However, as far as we are aware, this aspect has not been explored to a great extent in the context of interacting with agents. The following sections present end-to-end pipelines that show promising performance in addressing each research question. All the baselines are made publicly available at [3]

In addition to the baselines discussed in the following sections, we ran initial experiments using Large Language Models that highlight their application in solving this task is not a straightforward endeavor. The grounded nature of the task poses challenges when directly employing LLMs. Our experiments have shown that the transformation of voxel world information into textual format and the subsequent prompt engineering required to address these tasks using LLMs can be a complex and resource-intensive process. We recognize the potential benefits of exploring the use of larger language models for this task, which aligns with our future research direction. Further details on employing LLMs for this task can be found in Appendix A.2.

5.1 When: Clarification Need Prediction

We report the performance of baselines in Tab. 4 and utilize the F-1 Score as the evaluation metric as it provides a balanced measure of precision and recall for this classification task of predicting ambiguity in instructions.

[3] https://bit.ly/3qZ7QQD

Table 4: Results of the baselines on *When* to ask clarifying questions.

Baseline	F-1 score
Fine-tuned BERT (Sec. 5.1.1)	0.732
Text-Grid Cross Modularity (Sec. 5.1.2)	0.757
Textual Grid world State (Sec. 5.1.3)	0.761

5.1.1 BERT fine-tuning

Due to the substantial amount of training data in our collected dataset, one straightforward baseline (Aliannejadi et al., 2021b) to determine whether an instruction requires a clarifying question would be fine-tuning LMs such as BERT (Devlin et al., 2018) followed by a classification layer. This approach has shown promising performance on similar classification tasks (Arabzadeh et al., 2022) demonstrated in Tab. 4.

5.1.2 Text-Grid Cross Modularity

This baseline (Shi et al., 2023) has shown improvement over the BERT fine-tuning approach (Sec. 5.1.1) and consists of the following major components: *1)Utterance Encoder*, where Architect and Builder annotations would be added before each architect utterance A_t and each builder utterance B_t, respectively. Then, the dialogue utterances are represented as $D_t = architectA_t \oplus builderB_t$ at the turn t, where $\oplus$ is the operation of sequence concatenation. The dialogue is encoded through pre-trained language models such as BERT. *2) World state encoder* aims to represent the pre-built structure using a voxel-based grid. Each grid state is encoded as a 7-dimensional one-hot vector, representing either an empty space or a block of one of six colors. This encoding results in a $7 \times 11 \times 9 \times 11$ representation of the world state. The structure of the World State Encoder is similar to the approach presented in (Jayannavar et al., 2020). It comprises 3D-convolutional layers followed by a Rectified Linear Unit (ReLU) activation function. This configuration allows the encoder to extract meaningful features from the voxel-based grid representation of the world state. By applying convolutional layers and non-linear activation, the World State Encoder captures spatial dependencies and abstract representations of the pre-built structure. *3) Fusion module* consists of three sub-modules: one Single-Modality and two Cross-Modality. The former modules are based on self-attention layers, and the latter on cross-attention layers. These take as input the world state representation and dialogue history representation. Between every successive pair of grid single-modality modules or text single-modality modules, there is a cross-modality module. *4) Linear projection layer*, this component contains one linear projection to obtain a scalar value for the final binary classification through the Sigmoid function. Finally, the combination of the four aforementioned components obtained F-1 score of 0.757 on the task of *When*. While this approach might seem like the model is deciding whether the follower needs to speak, it aligns with the setup where the agent must decide whether to ask clarifying questions or to act on the most likely action that might lead to a successful task completion.

5.1.3 Text-Grid World State

This baseline focuses on mapping the GridWorld state to a textual context, which is then added as a prefix to the verbalizations of the Architect-Agent. This approach utilizes an automated description of the number of blocks per color in the pre-built structures. For instance, a voxel world can be automatically converted into a textual description like 'There are 4 levels. There are 15 different blocks. At level 0, there are 3 green blocks. Above the 1st level, there are 2 purple, 2 yellow, and 1 green block. Above level 2, there are 3 green blocks. Above the 3rd level, there are 2 yellow and 2 green blocks.' This description provides important contextual information about the voxel world and contributes to the improved performance of the simple LLM fine-tuning baseline. We note that the proposed approach could be applied to fine-tune any widely used Language Model such as BERT. However, the reported performance was achieved using Deberta-v3-base[4]. Overall, including a textual description of the voxelworld has enhanced the simple LM fine-tuning baseline by 4% in terms of performance (Tab. 4). This approach showcases the importance of incorporating relevant contextual information to enhance the understanding and classification of language-guided collaborative tasks.

5.2 What: Clarifying Question Retrieval

What to ask as clarifying questions has shown to be quite a challenging task (Aliannejadi et al., 2019). As such, similar to (Aliannejadi et al., 2020a, 2021b), we simplify the task by ranking a pool of clarifying questions based on their relevance to ambiguous instructions to place the most pertinent clarifying questions at the top of the ranked list. At inference time, the pool was designed to include all clarifying questions in the test set. Given that the relevance judgments for this task are sparse. Namely, only one clarifying question per ambiguous instruction is annotated. We evaluate the task using the Mean Reciprocal Rank (MRR) at cut-off 20. This evaluation approach is consistent with well-known benchmarks like MS MARCO (Nguyen et al., 2016). Tab. 5 presents the performance of BM25 (Robertson et al., 2009, 1995), which is a widely used and well-known ranking function used for Information Retrieval, followed by the two introduced baselines, measured using the MRR@20.

[4]https://bit.ly/3TldRTY

Table 5: Performance of the baselines on *What* to ask as clarifying question.

Baseline	MRR@20
BM25	0.3410
Text-World Fusion Ranker (5.2.1)	0.5360
State-Instruction Concatenation Ranker (5.2.2)	0.5960

5.2.1 Text-World Fusion Ranker

In this baseline, the instruction and the state of the voxel world are represented individually as text representation and world representation, respectively. Further, the encoded text representation and the world representation are concatenated, and the vector is passed through a two-layer MLP to obtain the final representation. The model is trained using a CrossEntropy loss function over 10-fold cross-validation. At inference time, the ensemble predictions of the 10 models are used for the final predictions. In the following, we elaborate on each of the text and world representations:

Text Representation (TR): A frozen DeBERTa-v3-base model has demonstrated promising performance for ranking. This baseline encodes the instructions, followed by a separator and a question. The last 4 layers of DeBERTa are concatenated and passed through a two-layer BiLSTM to acquire TR.

World Representation (WR): WR is utilized to create a 3D grid. The 3D grid represents a three-dimensional matrix representing the voxel world environment. Each block within this grid is represented by a specific number corresponding to different colors in the matrix. This is subsequently passed through a 1D convolutional network to simplify the height dimension (y), and then the resulting vector is passed through a 2D convolutional network to reduce the width/length (x, z) dimensions. The underlying assumption is that height occupies a different semantic space from the interchangeable x, z dimensions. For example, an instruction might include references to a *tower* or *column*, which would be a stack of objects in the y direction, while a *wall* could extend in the x or z direction. Ultimately, the size of the 3D grid is reduced by an AvgPooling layer to a 1D vector. This assumption is essentially made to make the 3D structure simplified into a 2D and then into a 1D representation to reduce the complexity of the representation. This simplification is akin to dimensionality reduction techniques and helps make the problem more manageable (Huang et al., 2022; Sainburg et al., 2020; Cao et al., 2018).

In addition, it has been revealed that certain straightforward post-processing tricks relying on certain assumptions about the content of questions given a world and instruction could be helpful. For example, the size of the ranking pool could be reduced by excluding questions that don't overlap with the given instructions. If the instruction doesn't mention a color like *blue*, and *blue* is also absent in the world, it can be assumed that the question will not reference the word *blue*. While these heuristic rules may seem somewhat aggressive, they have proven useful in excluding additional questions irrelevant to the instruction, as we see that Text-World Fusion Ranker utilized these approaches.

5.2.2 State-Instruction Concatenation Ranker

To comprehend the concept of relevance, the approach of aligning queries and relevant items closely in embedding space while distancing queries from irrelevant items in the same space has proven to be effective (Izacard et al., 2021; Reimers and Gurevych, 2019; Karpukhin et al., 2020; Zhan et al., 2021). Similarly, in this baseline, each positive question is paired with sampled irrelevant negative questions drawn from the candidate questions. The similarity between the instruction and the question is then measured using a BERT-like pre-trained LM.

To include information from the world state and pre-built structure, state information, such as the colors and numbers of initialized blocks, is encoded in natural language and then concatenated with the instruction. It has been shown that clarifying questions about the same instruction can differ based on the world states (Shi et al., 2022; Aliannejadi et al., 2019; Deng et al., 2023). To avoid redundant state information and improve the model's generalization, randomly selecting only one color type of block as the state information has proven helpful. The state information and raw instruction are then concatenated and labeled with the keywords *state* and *instruction*, respectively. For example, the input could be: *state*: There are 9 green blocks; *instruction*: put a green block on top of the yellow and the two blue ones. To balance the data distribution Easy Data Augmentation (EDA) has been adapted (Wei and Zou, 2019), which could expand the dataset by synonym replacement, random insertion, random swap, and random deletion, according to a pre-defined ratio. Moreover, taking inspiration from DAPT (Gururangan et al., 2020), datasets such as (Kiseleva et al., 2022b; Narayan-Chen et al., 2019b; Shi et al., 2022; Zholus et al., 2022) are used for performing domain-adaptive fine-tuning. Further, we propose to use the Fast Gradient Method (FGM), inspired by adversarial training, to mitigate the overfitting problem (Goodfellow et al., 2014). Finally, taking cues from (Gao et al., 2021), the list-wise loss is used to train the model.

6 Conclusions and Future Work

In conclusion, our paper addresses the crucial issue of enabling natural interaction in grounded human-AI agent collaboration. We achieve this by allowing agents to clarify instructions through a familiar and friendly interface, such as the use of clarifying questions. A significant obstacle hindering progress in this field has been the scarcity of appropriate datasets and scalable, extensible data collection tools. To address this challenge, we developed a crowdsourcing tool specifically designed for collecting interactive grounded language instructions within a Minecraft-like environment at a large scale. We created a dataset of human-to-human grounded language instructions, accompanied by clarifying questions which could be useful for a wide range of natural language understanding and reinforcement learning tasks. Furthermore, we established baselines for predicting clarifying questions, providing a benchmark for evaluating the performance of future models and algorithms in this domain. As future work, we plan to investigate how LLMs can be applied for our task. We also plan to develop an evaluation framework that incorporates human judgments or task-specific metrics to provide a better understanding of the performance and limitations of the proposed methods. Additionally, we plan to conduct comprehensive user studies to evaluate the usability of the generated clarifying questions in real-world scenarios. We anticipate that exploring these future directions will contribute to even a greater understanding of the challenges and potential solutions involved generating clarifying questions for instruction based interactions.

7 Limitations

Our paper centers on the utilization of a Minecraft-like environment to examine human-AI interaction. While this emphasis may not comprehensively encapsulate the intricacies of real-world scenarios, it affords the opportunity to scrutinize specific facets of the problem in an isolated and safe environment. Nevertheless, there are constraints within the task scenarios, including the consideration of potential variations in task complexity. This may constrain the understanding of how the generation of clarifying questions may vary in different contexts. Consequently, the generalizability and applicability of our findings to real-world settings may be influenced by these factors. However, we believe the suggested environment and the data collection tool allow exploration for further scenarios.

Moreover, the paper relies on a crowdsourcing tool for data collection, which introduces the possibility of biases in the dataset. The demographic composition, skill levels, and motivations of the crowd workers may impact the quality and representativeness of the collected data. To mitigate these biases, we introduced sophisticated training and tests for crowd-source workers to enable them to complete tasks. To address any potential ethical issues, all the crowd-source workers signed an IRB.

References

Josh Abramson, Arun Ahuja, Iain Barr, Arthur Brussee, Federico Carnevale, Mary Cassin, Rachita Chhaparia, Stephen Clark, Bogdan Damoc, Andrew Dudzik, et al. 2020. Imitating interactive intelligence. *arXiv preprint arXiv:2012.05672*.

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. 2020a. Convai3: Generating clarifying questions for open-domain dialogue systems (clariq).

Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. 2020b. Convai3: Generating clarifying questions for open-domain dialogue systems (clariq). *arXiv preprint arXiv:2009.11352*.

Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. 2021a. Building and evaluating open-domain dialogue corpora with clarifying questions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4473–4484, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeffrey Dalton, and Mikhail Burtsev. 2021b. Building and evaluating open-domain dialogue corpora with clarifying questions. *arXiv preprint arXiv:2109.05794*.

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval*, pages 475–484.

Negar Arabzadeh, Mahsa Seifikar, and Charles LA Clarke. 2022. Unsupervised question clarity prediction through retrieved item coherency. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 3811–3816.

Cristian-Paul Bara, Sky CH-Wang, and Joyce Chai. 2021. Mindcraft: Theory of mind modeling for situated dialogue in collaborative tasks. *arXiv preprint arXiv:2109.06275*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Yonatan Bisk, Deniz Yuret, and Daniel Marcu. 2016. Natural language communication with robots. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 751–761.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Mikhail Burtsev, Aleksandr Chuklin, Julia Kiseleva, and Alexey Borisov. 2017. Search-oriented conversational ai (scai). In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, pages 333–334.

Zhihao Cao, MU Shaomin, XU Yongyu, and Mengping Dong. 2018. Image retrieval method based on cnn and dimension reduction. In *2018 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, pages 441–445. IEEE.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. 2023. Grounding large language models in interactive environments with online reinforcement learning. *arXiv preprint arXiv:2302.02662*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Edgar F Codd. 1974. *Seven steps to rendezvous with the casual user*. IBM Corporation.

Ann Copestake and Karen Sparck Jones. 1990. Natural language interfaces to databases.

Yang Deng, Shuaiyi Li, and Wai Lam. 2023. Learning to ask clarification questions with spatial reasoning. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2113–2117.

Aditya Desai, Sumit Gulwani, Vineet Hingorani, Nidhi Jain, Amey Karkare, Mark Marron, Subhajit Roy, et al. 2016. Program synthesis using natural language. In *Proceedings of the 38th International Conference on Software Engineering*, pages 345–356. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. 2020. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition*, pages 187–208. Springer, Cham.

Ahmed Elgohary, Saghar Hosseini, and Ahmed Hassan Awadallah. 2020. Speak to your parser: Interactive text-to-SQL with natural language feedback. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2065–2077, Online. Association for Computational Linguistics.

Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. 2022. Minedojo: Building open-ended embodied agents with internet-scale knowledge.

Ethan Fast, Binbin Chen, Julia Mendelsohn, Jonathan Bassen, and Michael S Bernstein. 2018. Iris: A conversational agent for complex tasks. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 473. ACM.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. Rethink training of BERT rerankers in multi-stage retrieval pipeline. *CoRR*, abs/2101.08751.

Kevin A Gluck and John E Laird. 2018. *Interactive task learning: Humans, robots, and agents acquiring new tasks through natural interactions*. The MIT Press.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Jonathan Gray, Kavya Srinet, Yacine Jernite, Haonan Yu, Zhuoyuan Chen, Demi Guo, Siddharth Goyal, C. Lawrence Zitnick, and Arthur Szlam. 2019a. Craftassist: A framework for dialogue-enabled interactive agents.

Jonathan Gray, Kavya Srinet, Yacine Jernite, Haonan Yu, Zhuoyuan Chen, Demi Guo, Siddharth Goyal, C. Lawrence Zitnick, and Arthur Szlam. 2019b. CraftAssist: A Framework for Dialogue-enabled Interactive Agents. *arXiv:1907.08584 [cs]*. ArXiv: 1907.08584.

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

William H Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, Cayden Codel, Manuela Veloso, and Ruslan Salakhutdinov. 2019. Minerl: A large-scale dataset of minecraft demonstrations. *arXiv preprint arXiv:1907.13440*.

Gary G Hendrix, Earl D Sacerdoti, Daniel Sagalowicz, and Jonathan Slocum. 1978. Developing a natural language interface to complex data. *ACM Transactions on Database Systems (TODS)*, 3(2):105–147.

Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, Marcus Wainwright, Chris Apps, Demis Hassabis, and Phil Blunsom. 2017. Grounded language learning in a simulated 3d world.

Dihe Huang, Ying Chen, Yikang Ding, Jinli Liao, Jianlin Liu, Kai Wu, Qiang Nie, Yong Liu, Chengjie Wang, and Zhiheng Li. 2022. Rethinking dimensionality reduction in grid-based 3d object detection. *arXiv preprint arXiv:2209.09464*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Prashant Jayannavar, Anjali Narayan-Chen, and Julia Hockenmaier. 2020. Learning to execute instructions in a minecraft dialogue. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 2589–2602.

Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. 2016. The malmo platform for artificial intelligence experimentation. In *IJCAI*, pages 4246–4247. Citeseer.

Anssi Kanervisto, Stephanie Milani, Karolis Ramanauskas, Nicholay Topin, Zichuan Lin, Junyou Li, Jianing Shi, Deheng Ye, Qiang Fu, Wei Yang, Weijun Hong, Zhongyue Huang, Haicheng Chen, Guangjun Zeng, Yue Lin, Vincent Micheli, Eloi Alonso, François Fleuret, Alexander Nikulin, Yury Belousov, Oleg Svidchenko, and Aleksei Shpilman. 2022. Minerl diamond 2021 competition: Overview, results, and lessons learned.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Julia Kiseleva, Ziming Li, Mohammad Aliannejadi, Shrestha Mohanty, Maartje ter Hoeve, Mikhail Burtsev, Alexey Skrynnik, Artem Zholus, Aleksandr Panov, Kavya Srinet, Arthur Szlam, Yuxuan Sun, Katja Hofmann, Marc-Alexandre Côté, Ahmed Awadallah, Linar Abdrazakov, Igor Churin, Putra Manggala, Kata Naszadi, Michiel van der Meer, and Taewoon Kim. 2022a. Interactive grounded language understanding in a collaborative environment: Iglu 2021. In *NeurIPS 2021 Competitions and Demonstrations Track*, pages 146–161. PMLR.

Julia Kiseleva, Ziming Li, Mohammad Aliannejadi, Shrestha Mohanty, Maartje ter Hoeve, Mikhail Burtsev, Alexey Skrynnik, Artem Zholus, Aleksandr Panov, Kavya Srinet, et al. 2021. Neurips 2021 competition iglu: Interactive grounded language understanding in a collaborative environment. *arXiv preprint arXiv:2110.06536*.

Julia Kiseleva, Alexey Skrynnik, Artem Zholus, Shrestha Mohanty, Negar Arabzadeh, Marc-Alexandre Côté, Mohammad Aliannejadi, Milagro Teruel, Ziming Li, Mikhail Burtsev, et al. 2022b. Interactive grounded language understanding in a collaborative environment: Retrospective on iglu 2022 competition. In *NeurIPS 2022 Competition Track*, pages 204–216. PMLR.

Julia Kiseleva, Kyle Williams, Ahmed Hassan Awadallah, Aidan C Crook, Imed Zitouni, and Tasos Anastasakos. 2016a. Predicting user satisfaction with intelligent assistants. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 45–54.

Julia Kiseleva, Kyle Williams, Jiepu Jiang, Ahmed Hassan Awadallah, Aidan C Crook, Imed Zitouni, and Tasos Anastasakos. 2016b. Understanding user satisfaction with intelligent assistants. In *Proceedings of the 2016 ACM on Conference on Human Information Interaction and Retrieval*, pages 121–130.

Arne Köhn, Julia Wichlacz, Christine Schäfer, Alvaro Torralba, Jörg Hoffmann, and Alexander Koller. 2020. Mc-saar-instruct: a platform for minecraft instruction giving agents. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 53–56.

Toby Jia-Jun Li, Tom Mitchell, and Brad Myers. 2020a. Interactive task learning from GUI-grounded natural language instructions and demonstrations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Ziming Li, Julia Kiseleva, and Maarten De Rijke. 2019. Dialogue generation: From imitation learning to inverse reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6722–6729.

Ziming Li, Julia Kiseleva, and Maarten de Rijke. 2020b. Rethinking supervised learning and reinforcement learning in task-oriented dialogue systems. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3537–3546, Online. Association for Computational Linguistics.

Ziming Li, Julia Kiseleva, and Maarten de Rijke. 2021. Improving response quality with backward reasoning in open-domain dialogue systems. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1940–1944.

Ziming Li, Sungjin Lee, Baolin Peng, Jinchao Li, Julia Kiseleva, Maarten de Rijke, Shahin Shayandeh, and Jianfeng Gao. 2020c. Guided dialogue policy learning without adversarial learning in the loop. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2308–2317, Online. Association for Computational Linguistics.

Bing Liu and Ian Lane. 2017. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 482–489. IEEE.

Bing Liu and Ian Lane. 2018. Adversarial learning of task-oriented neural dialog models. In *Proceedings of the SIGDIAL 2018 Conference*, pages 350–359.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta:

A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Nikhil Mehta, Milagro Teruel, Patricio Figueroa Sanz, Xin Deng, Ahmed Hassan Awadallah, and Julia Kiseleva. 2023. Improving grounded language understanding in a collaborative environment by interacting with agents through help feedback. *arXiv preprint arXiv:2304.10750*.

Koh Mitsuda, Ryuichiro Higashinaka, Yuhei Oga, and Sen Yoshida. 2022. Dialogue collection for recording the process of building common ground in a collaborative task. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5749–5758, Marseille, France. European Language Resources Association.

Shrestha Mohanty, Negar Arabzadeh, Milagro Teruel, Yuxuan Sun, Artem Zholus, Alexey Skrynnik, Mikhail Burtsev, Kavya Srinet, Aleksandr Panov, Arthur Szlam, Marc-Alexandre Côté, and Julia Kiseleva. 2022. Collecting interactive multi-modal datasets for grounded language understanding. *arXiv preprint arXiv:2211.06552*.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019a. Collaborative dialogue in minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019b. Collaborative dialogue in Minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415, Florence, Italy. Association for Computational Linguistics.

Clifford Nass and Youngme Moon. 2000. Machines and mindlessness: Social responses to computers. *Journal of social issues*, 56(1):81–103.

Khanh Nguyen and Hal Daumé III au2. 2019. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning.

Khanh Nguyen, Yonatan Bisk, and Hal Daumé III au2. 2022. A framework for learning to request rich and contextually useful information from humans.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. *choice*, 2640:660.

Haruna Ogawa, Hitoshi Nishikawa, Takenobu Tokunaga, and Hikaru Yokono. 2020. Gamification platform for collecting task-oriented dialogue data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 7084–7093, Marseille, France. European Language Resources Association.

OpenAI. 2023. Gpt-4 technical report.

Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gökhan Tür, and Dilek Hakkani-Tür. 2022. Teach: Task-driven embodied agents that chat. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 2017–2025. AAAI Press.

Joon Sung Park, Joseph C O'Brien, Carrie J Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. *arXiv preprint arXiv:2304.03442*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Allen Z. Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. 2023. Robots that ask for help: Uncertainty alignment for large language model planners.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Tim Sainburg, Marvin Thielk, and Timothy Q Gentner. 2020. Finding, visualizing, and quantifying latent structure across diverse animal vocal repertoires. *PLoS computational biology*, 16(10):e1008228.

Zhengxiang Shi, Yue Feng, and Aldo Lipani. 2022. Learning to execute actions or ask clarification questions. *arXiv preprint arXiv:2204.08373*.

Zhengxiang Shi, Jerome Ramos, To Eun Kim, Xi Wang, Hossein A. Rahmani, and Aldo Lipani. 2023. When and what to ask through world states and text instructions: Iglu nlp challenge solution.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2019. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. *CoRR*, abs/1912.01734.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749.

Alexey Skrynnik, Zoya Volovikova, Marc-Alexandre Côté, Anton Voronov, Artem Zholus, Negar Arabzadeh, Shrestha Mohanty, Milagro Teruel, Ahmed Awadallah, Aleksandr Panov, Mikhail Burtsev, and Julia Kiseleva. 2022. Learning to solve voxel building embodied tasks from pixels and natural language instructions. *arXiv preprint arXiv:2211.00688*.

Kavya Srinet, Yacine Jernite, Jonathan Gray, and Arthur Szlam. 2020. CraftAssist instruction parsing: Semantic parsing for a voxel-world assistant. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4693–4714, Online. Association for Computational Linguistics.

Yu Su, Ahmed Hassan Awadallah, Madian Khabsa, Patrick Pantel, Michael Gamon, and Mark Encarnacion. 2017. Building natural language interfaces to web apis. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 177–186. ACM.

Alane Suhr, Claudia Yan, Jacob Schluger, Stanley Yu, Hadi Khader, Marwa Mouallem, Iris Zhang, and Yoav Artzi. 2019. Executing instructions in situated collaborative interactions. *CoRR*, abs/1910.03655.

Arthur Szlam, Jonathan Gray, Kavya Srinet, Yacine Jernite, Armand Joulin, Gabriel Synnaeve, Douwe Kiela, Haonan Yu, Zhuoyuan Chen, Siddharth Goyal, Demi Guo, Danielle Rothermel, C. Lawrence Zitnick, and Jason Weston. 2019. Why Build an Assistant in Minecraft? *arXiv:1907.09273 [cs]*. ArXiv: 1907.09273.

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2019. Vision-and-dialog navigation. *CoRR*, abs/1907.04957.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023a. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Sida I. Wang, Samuel Ginn, Percy Liang, and Christopher D. Manning. 2017. Naturalizing a programming language via interactive learning. *CoRR*, abs/1704.06956.

Sida I. Wang, Percy Liang, and Christopher D. Manning. 2016. Learning language games through interaction. *CoRR*, abs/1606.02447.

Xin Wang, Taein Kwon, Mahdi Rad, Bowen Pan, Ishani Chakraborty, Sean Andrist, Dan Bohus, Ashley Feniello, Felipe Vieira Frujeri, Neel Joshi, and Marc Pollefeys. 2023b. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. In *ICCV 2023*.

Jason W. Wei and Kai Zou. 2019. EDA: easy data augmentation techniques for boosting performance on text classification tasks. *CoRR*, abs/1901.11196.

Terry Winograd. 1972. Understanding natural language. *Cognitive psychology*, 3(1):1–191.

W. A. Woods, Ronald M Kaplan, and Bonnie L. Webber. 1972. The lunar sciences natural language information system: Final report. *BBN Report 2378*.

Ziyu Yao, Yu Su, Huan Sun, and Wen-tau Yih. 2019. Model-based interactive semantic parsing: A unified framework and a text-to-SQL case study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5447–5458, Hong Kong, China. Association for Computational Linguistics.

Ziyu Yao, Yiqi Tang, Wen-tau Yih, Huan Sun, and Yu Su. 2020. An imitation game for learning semantic parsers from user interaction.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1503–1512.

Yi Zhang, Sujay Kumar Jauhar, Julia Kiseleva, Ryen White, and Dan Roth. 2021. Learning to decompose and organize complex tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2726–2735, Online. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

Artem Zholus, Alexey Skrynnik, Shrestha Mohanty, Zoya Volovikova, Julia Kiseleva, Artur Szlam, Marc-Alexandre Coté, and Aleksandr I Panov. 2022. Iglu gridworld: Simple and fast environment for embodied dialog agents. *arXiv preprint arXiv:2206.00142*.

A Appendix

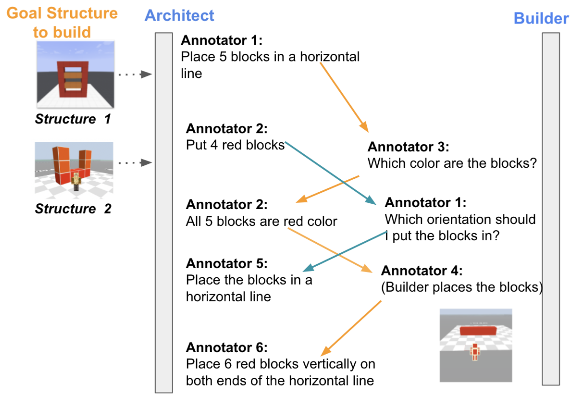


Figure 2: Example of multi-turn data collection, where the Architect can see the goal structure and provides instructions for the Builder. The blue arrows indicate turns for the first goal structure, the orange arrows indicate turns for the second goal structure. Annotators can switch roles between architect and builder for different structures.

A.1 Data Collection Details

Multi-turn Data Collection: In Figure 2, we illustrate an example of multi-turn data collection. In this scenario, the Architect can observe the goal structure and offer instructions to the Builder. The blue arrows represent the turns associated with the first goal structure, while the orange arrows correspond to the turns related to the second goal structure. Annotators can switch roles between architect and builder for different structures. Fig. 2 illustrates this concept of our data collection methodology with different annotators (1, 3, 2, 4, and 6) collaborating to construct Structure 1. Annotators can switch roles between architect and builder for different structures.

Fig. 3 illustrates the overall design of the tool. Our tool can be integrated with crowd-sourcing platforms to provide an interface for participants to complete tasks. Fig. 5 demonstrates MTurk views of the Data Collection Tool (Sec.3) for the Multi-Turn Dataset (Sec.4.1). We have the Architect Task, where the Architect provides instructions to the Builder based on the provided target structure. Next, we have the Builder Task, where instructions and the current structure built so far are shown. The Builder can mark the instructions as unclear or will follow the instructions by adjusting blocks in the voxelworld. Finally, we have the Intermediate Architect Task, where the Architect is shown the

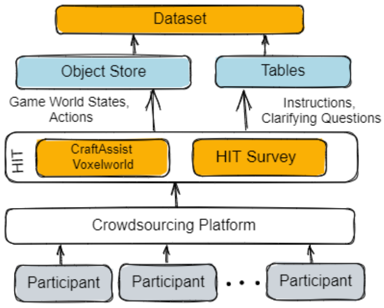


Figure 3: The architecture of the data crowdsourcing collection tool

progress of the structure built so far and provides the next instruction.

Examples of Single-Turn Dataset: Tab. 6 provides examples of instructions marked as unclear in the Single-Turn Dataset along with different kinds of clarifying questions posed by annotators (Sec.4.2). Clarifying questions consist of topics such as color, direction, and identification of blocks.

A.2 Large Language Models as baselines

In our earlier discussion in Section 5, we highlighted that applying Large Language Models (LLMs) to the task of determining when and what to ask as clarifying questions in our designed environment is not a straightforward process. This complexity arises primarily due to the multimodal nature of the task and the significant engineering efforts required to create effective prompts. While we do not suggest that leveraging LLMs for this task is impossible, it is important to clarify that our paper's primary focus lies in benchmarking and dataset creation. Integrating LLMs into our study falls beyond the scope of this research.

Additionally, We address the challenge of determining when and what clarifying questions to ask by employing a combination of classification and retrieval methods instead of relying on text generation. Our decision was influenced by several factors, including the absence of well-established evaluation procedures for LLM text generation and the need to handle complex structures like action states of the world, which serve as inputs to our current pre-trained model baselines. Nonetheless, we conducted preliminary experiments using GPT-3.5-Turbo to explore their potential applicability to this task.

In these experiments, we randomly selected 50 instructions and utilized their previous utterances as information to reconstruct the pre-built structure. We then prompted the LLM to determine whether,

Table 6: Examples of Unclear Instructions and Clarifying Questions

Instruction	Clarifying Question
Place four blocks to the east of the highest block, horizontally.	Which color blocks?
Destroy 2 purple blocks and then build 3 green blocks diagonally.	Which two purple blocks need to be destroyed?
Destroy the 3 stacked red blocks on the east side. Replace them with 3 stacked blue boxes	Which three of the four stacked red blocks on the east side need to be destroyed?
Make a rectangle that is the width and height of the blue shape and fill it in with purple blocks.	Which side I need to make the rectangle is not clear
Facing South remove the rightmost purple block. Place a row of three orange blocks to the left of the upper leftmost purple block. Place two orange blocks above and below the leftmost orange block.	Which one of the rightmost blocks should be removed?
Facing north and purple-green blocks will be arranged one by one.	Where would you like to place the purple and green blocks exactly?

You are participating in a game set in a Minecraft-like world. In this game, there are two roles: the Architect and the Builder.
1. The Architect: This player provides instructions for building structures in the game environment.
2. The Builder: This player's role is to follow the instructions given by the Architect and execute them within the game environment.
During the game, the Builder has two response options:
- If the instruction provided by the Architect is clear and can be executed without any need for further clarification or questions, the Builder responds with "yes. The instruction is clear."
- If the instruction is unclear or requires clarification from the Architect before it can be executed, the Builder responds with "no" and generates a clarification question.
reply only a "Yes. The instruction is clear" or a "No" followed by a relevant clarification question.

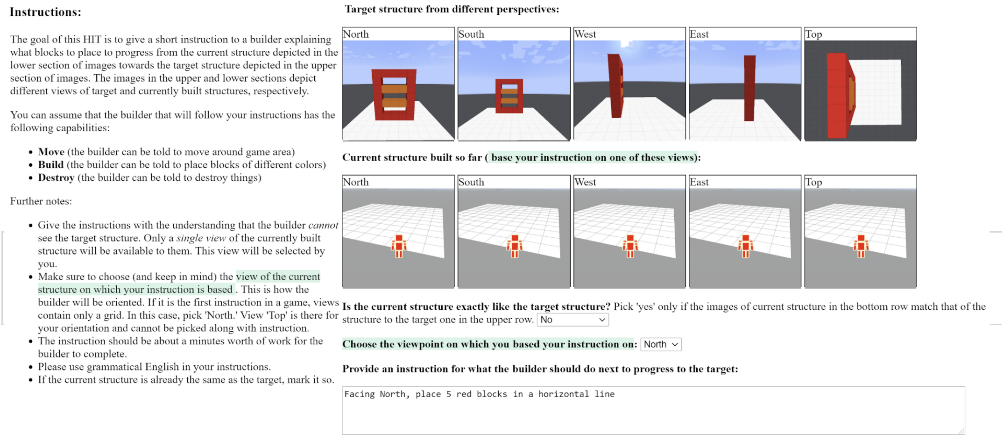
Previous Dialogue: <Architect> Facing North, Build a blue block in the left most corner.
<Architect> Destroy the blue block and build a purple block there.
<Architect> Facing East, place one green block on the very top right corner of the map.
Starting Grid world of 3D blocks in the format $(x, y, z, color)$: $(-5, -1, -5, purple)(-5, 9, 5, 'green')$

Current Instruction: <Architect> In the northeast corner place one blue block. In the southwest corner place one purple block then a red block on top of that.

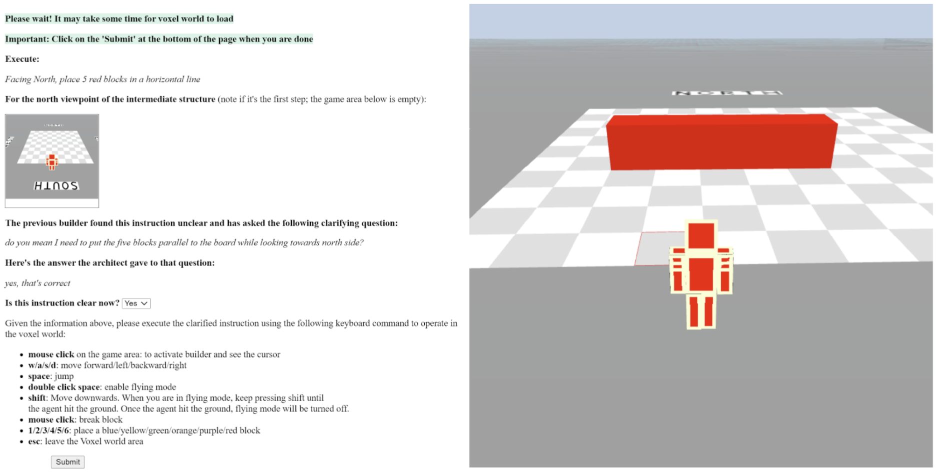
Figure 4: Example of using LLMs for solving the clarifying question need task.

given the pre-existing instructions from the Architect, the new instruction was clear or if the builder needed to pose clarifying questions. An example of such a prompt can be found in Figure 4, focusing on the "when to ask clarifying questions" task. However, the results were far from satisfactory when compared to the baseline models, yielding an F1 score of 0.45, which was significantly lower than the F1 scores achieved by our baseline models, as reported in Table 4. All baseline models achieved F1 scores above 0.732.

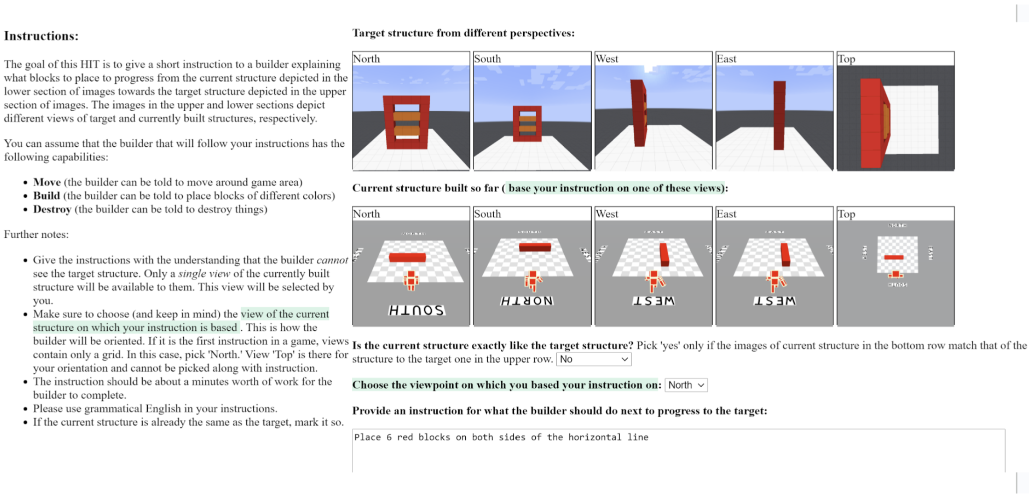
We believe that the performance of LLMs can be enhanced through improved prompt engineering and a better representation of the voxel world. However, we decided not to include these findings in the paper to avoid potential misinterpretations. Our primary aim was to establish a benchmark with clear and reproducible baselines. While we acknowledge the potential of LLMs for this task, we consider this aspect as part of our future work, which extends beyond the scope of the current study.



(a) Architect Task



(b) Builder Task



(c) Intermediate Structure Architect Task

Figure 5: View of MTurk Data Collection Tool