Enabling Robots to Understand Incomplete Natural Language Instructions Using Commonsense Reasoning

Haonan Chen, Hao Tan, Alan Kuntz, Mohit Bansal, and Ron Alterovitz

Department of Computer Science
University of North Carolina at Chapel Hill
Chapel Hill, NC 27599, USA
{haonanchen, haotan, adkuntz, mbansal, ron}@cs.unc.edu

***Abstract*— Enabling robots to understand instructions provided via spoken natural language would facilitate interaction between robots and people in a variety of settings in homes and workplaces. However, natural language instructions are often missing information that would be obvious to a human based on environmental context and common sense, and hence does not need to be explicitly stated. In this paper, we introduce Language-Model-based Commonsense Reasoning (LMCR), a new method which enables a robot to listen to a natural language instruction from a human, observe the environment around it, and automatically fill in information missing from the instruction using environmental context and a new commonsense reasoning approach. Our approach first converts an instruction provided as unconstrained natural language into a form that a robot can understand by parsing it into verb frames. Our approach then fills in missing information in the instruction by observing objects in its vicinity and leveraging commonsense reasoning. To learn commonsense reasoning automatically, our approach distills knowledge from large unstructured textual corpora by training a language model. Our results show the feasibility of a robot learning commonsense knowledge automatically from web-based textual corpora, and the power of learned commonsense reasoning models in enabling a robot to autonomously perform tasks based on incomplete natural language instructions.**

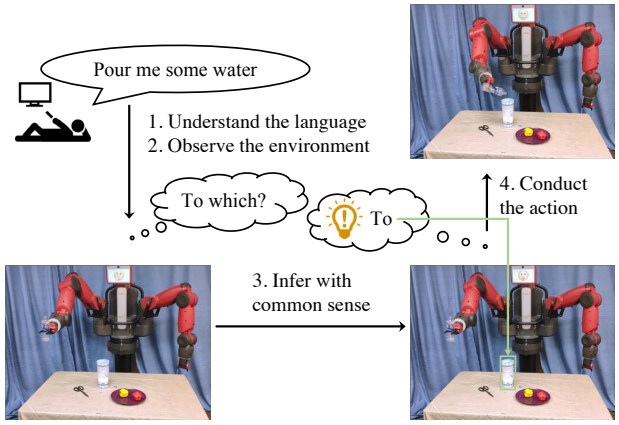


Fig. 1 Natural language instructions are often incomplete. An example of a natural language controlled robot employing commonsense knowledge to interpret an instruction with missing information. A person gives an instruction "pour me some water" but the robot cannot carry out the action without knowing where to pour. After scanning the environment, the robot uses commonsense knowledge to determine the missing parameters and successfully perform the action.

I. Introduction

Natural language has the potential to provide a powerful and intuitive interface for people to provide instructions to robots in a variety of settings, from homes to workplaces. However, natural language is inherently unstructured and often reliant on environmental context, which makes it challenging for robots to correctly and precisely interpret natural language. Consider a scenario in a home setting in which the robot is holding a bottle of water and there are scissors, a plate, some bell peppers and a cup on a table (see Fig. 1). A human gives an instruction, *"pour me some water"*, to the robot. From the robot's perspective this instruction is incomplete. It contains missing information that the robot must figure out, namely the object that the water should be poured into. A robot that has the common sense to automatically resolve such incompleteness in natural language instructions, just as humans do intuitively, will allow humans to interact with it more naturally and increase its overall usefulness. To this end, we introduce Language-Model-based Commonsense Reasoning (LMCR), a new approach which enables a robot to listen to a natural language instruction from a human, observe the environment around it, automatically resolve missing information in the instruction, and then autonomously perform the specified task.

The premise of LMCR is that it automatically fills in missing information from on instruction based on the following principle: people are more likely to omit information from an instruction if it is obvious to the listener. If a human gives the instruction *"pour me some water"*, there may be no need to specify the object to pour the water into if, as in the example above, there is only one object available into which water is typically poured, namely, the cup. Inspired by this, we propose to use a neural network based *language model*, which acts as a probability distribution over sequences of words. At a high level, LMCR identifies missing information in the instruction, uses the robot's observations of its environment to generate candidate instructions that fill in the missing information (which represent potential resolutions of the incompleteness), and then uses the language model to rank the candidate instructions by their likelihoods. By assigning higher probabilities to candidate instructions that correspond

to more common tasks, the language model enables the robot to automatically fill in missing information in natural language instructions.

When a robot using LMCR receives an instruction provided as unstructured natural language, it must first determine what required information (if any) is missing in the instruction. To do this, LMCR first parses a natural language instruction (i.e., a sequence of words) into a *verb frame*, which is a structured representation of the instruction. A verb frame is a set containing (1) a predicate (i.e., a verb or verb phrase) and (2) a set of semantic roles and their associated content (i.e., the underlying relationships that words or phrases have with the predicate) [1]. For example, LMCR automatically parses the instruction *"pour me some water"* to the verb frame (pour, `Theme`: water, `Destination`: `?`), where "pour" is the predicate, "water" and "`?`" are arguments which are the expression that helps complete the meaning of a predicate, and "`Theme`" and "`Destination`" are semantic roles which specify the underlying relationship between arguments and the predicate. The empty tag `?` indicates that the argument of `Destination` is missing. The core of the process of parsing natural language instructions into verb frames is semantic role labeling (SRL), which involves predicting a label for every word in a sentence, and has been widely studied in natural language processing [2], [3], [4], [5], [6]. In LMCR, we use an off-the-shelf SRL model built by He et al. [6] for parsing sentences to verb frames.

After identifying what information (if any) is missing from an instruction, LMCR combines the robot's observations of the environment with a language model to fill in the missing information. We assume the robot has a computer vision system (e.g., using data from an RGB-D sensor with appropriate computer vision software) that enables it to detect relevant objects in its environment. Using the premise that missing information is likely omitted because it is obvious, LMCR fills in missing arguments in the parsed verb frame with objects detected in the environment. This yields several complete, candidate verb frames (where the number of candidates depends on the number of objects in the robot's environment). From each of these candidate verb frames, we generate a complete candidate natural language instruction. We then employ the trained language model to assign a probability to each candidate natural language instruction. Our language model makes use of a Recurrent Neural Network (RNN) [7], which we trained on large textual corpora of detailed task instructions. In this work we primarily focus on home assistance tasks. As such, we use the YouCook2 [8] and Now You're Cooking (NYC) [9] datasets as training corpora. After the language model has assigned a probability to each candidate natural language instruction, LMCR chooses the candidate with the highest probability as the complete description of the desired task. With all information in the verb frame now known, the robot can then execute a motion planner to accomplish the task in the completely-specified verb frame.

Prior work at the intersection of robotics and natural language processing has extensively explored the grounding of objects in natural language (e.g., identifying which object in the physical world is being referred to in natural language) [10], [11], [12], [13], [14], [15], [16], [17]. Our objective is not to ground objects that are stated in a sentence, but rather to intuit objects the speaker implied but did not explicitly state, essentially grounding unstated concepts in natural language instructions. A related problem studied widely in psycholinguistics is thematic fit and selectional preference modeling [18], [19], [20], [21], [22], [23], [24], [25], which aims at determining how likely a noun can be filled into a role of a predicate. One popular approach is to learn a "prototype embedding" for each predicate and role pair, which can be computed by averaging the word embeddings [26], [27] of several typical role fillers. By contrast, our approach builds a model on the entire sentence, not a specific role, a concept not considered in previous literature. This make it easy to extend our model to a broader range of predicates and roles since we do not have to compute a separate representation for each of them.

We provide an overview of the components of a robot using LMCR in Fig. 2. A robot using LMCR is equipped with "ears" and "eyes". The robot listens to the human instruction via a microphone and converts the human speech into a sequence of words using an automatic Speech Recognition module. The robot also observes the environment and identifies objects in the robot's vicinity (e.g., via an RGB-D camera and using computer vision detectors to find objects in the input camera images). The instruction and list of objects are fed to LMCR, which includes 2 modules. The first is the Predicate-Argument Parsing module, which parses the instruction to a verb frame using standard natural language processing tools. The verb frame, together with the list of sensed objects in the robot's vicinity, is sent to LMCR's second module, the Language Model Reasoning module, which forms the bulk of our contribution. This module fills the missing information in a verb frame with detected objects to create complete candidate verb frames, ranks the candidates with a neural network language model, and returns the candidate with the highest probability. LMCR's output is a complete verb frame with no missing information, which is passed to the Motion Planning module. The Motion Planning module begins by localizing in 3D space (relative to the robot) the relevant objects corresponding to arguments in the complete verb frame (e.g., the cup corresponding to the destination of the water in the example in Fig. 1). In our implementation of the Motion Planning module, we created a task-specific motion planner for each verb (e.g., pour) that the robot should be capable of executing. The motion planner computes feasible motions that avoid obstacles and move the end-effector to task-specific milestones that are defined with respect to the real-world 3D positions of the relevant objects from the complete verb frame. We use the motion planning toolkit MoveIt! [28] to compute a motion plan for the robot that reaches each milestone to accomplish the task.

To evaluate LMCR quantitatively, we collected a human-annotated dataset focused specifically on commonsense reasoning for instructions relevant to robots performing home

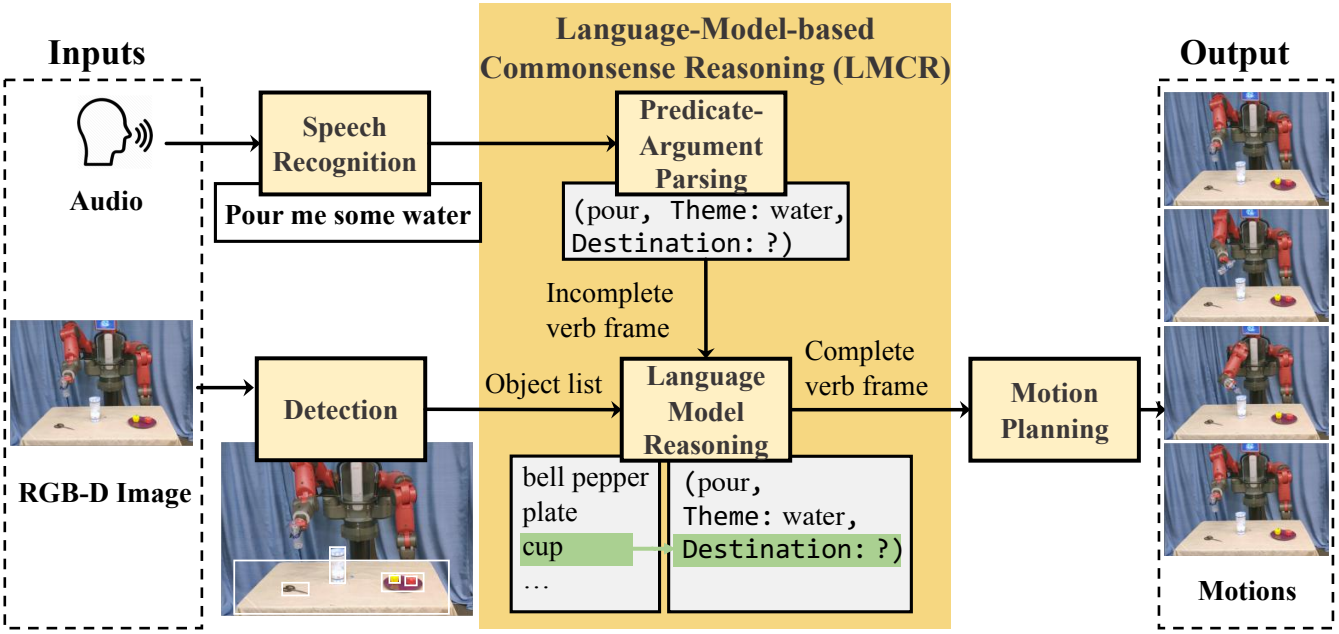


Fig. 2 The components of a robot with LMCR. A robot with LMCR is equipped with "ears" and "eyes". It listens to a human instruction via a microphone and converts it into a sequence of words with a Speech Recognition module. The robot also observes the environment to identify objects in the robot's vicinity (e.g., via an RGB-D camera and using a computer vision detector). The instruction and list of objects are fed to our new method LMCR, which consists of two modules. LMCR's Predicate-Argument Parsing module parses the instruction to a verb frame, which may contain missing information. LMCR's Language Model Reasoning module fills the missing information in a verb frame with detected objects to create complete candidate verb frames, ranks the candidates with a neural network language model, and returns the candidate with the highest probability. LMCR's output is a complete verb frame with no missing information, which is passed to the Motion Planning module and executed on the robot.

assistance tasks. Existing datasets on commonsense reasoning such as [29], [30], [31] consist mostly of general-purpose verbs and nouns and are not aimed specifically at robot manipulation applications, making them unsuitable for evaluating our method. Our dataset contains a set of "scenarios", each consisting of an incomplete instruction and a list of objects. For each object, we tasked humans with identifying how well the object would fit in the missing role of the instruction. We evaluate LMCR's prediction of the missing role against human judgments, which is the practice of previous research in thematic fit and selectional preference modeling [20], [21], [22], [23], [24], [25]. We also deploy our commonsense reasoning approach on a real, physical robot to evaluate its efficacy in the physical world, These experiment show the potential of LMCR to enable home assistance robots to achieve a greater level of autonomy by correctly interpreting spoken human instructions that may have missing information, as well the potential to enable more natural human interaction with the robot, allowing a human to speak to the robot as he or she would speak to another person.

II. Methods

A. Verb Frames

We first formally define verb frames, which are used throughout this work to represent the semantic nature of verbs and objects, including missing information, in a structured way. Following the notation in the frame semantic parsing [32], [33] and thematic fit evaluation [19] literature, we define a verb frame as $f = (v, r_1, a_1, \ldots, r_n, a_n)$, where v denotes the predicate, r_i and a_i denotes the i-th role and its argument respectively. In our work, r_i are drawn from a fixed, pre-defined set of role labels and are a function of the predicates. Each a_i is drawn from a fixed vocabulary $\mathcal{A}$ (note that $\mathcal{A}$ includes an empty tag `?` $\in \mathcal{A}$ indicating a missing argument).

In the evaluation of this work, we consider the circumstance where the human instruction is parsed into a verb frame with one role filled and one missing. We do not evaluate instructions that take more than 2 arguments, or instructions where either none or both of the arguments are missing. Thus, we denote a verb frame as $f = (v, r_1, a_1, r_2, a_2)$. We consider the following frame structures in our experiment (in format $v/r_1/r_2$):

(i) blend/`Patient`/`Co-Patient`
(ii) brush/`Theme`/`Destination`
(iii) dip/`Destination`/`Theme`
(iv) dump/`Theme`/`Destination`
(v) fill/`Destination`/`Theme`
(vi) fry/`Patient`/`Instrument`
(vii) heat/`Patient`/`Instrument`
(viii) pour/`Theme`/`Destination`
(ix) rub/`Theme`/`Destination`
(x) season/`Destination`/`Theme`
(xi) sprinkle/`Theme`/`Destination`

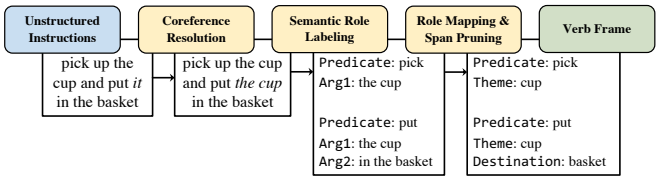


Fig. 3 Predicate-argument parsing process and example.

B. System Components for a LMCR-enabled Robot

A robot with LMCR takes as input a spoken instruction and a raw RGB-D image of the environment (see Fig. 2). The robot first parses the input instruction and detects objects in the environment. Using LMCR, the robot resolves incompleteness in the instructions and outputs a complete verb frame. The robot then computes a sequence of movements to execute the task specified in the complete verb frame. Details of each of the robot's modules are introduced below, and we cover the components of LMCR (Predicate-Argument Parsing and Language Model Reasoning) in the greatest depth.

a) Speech Recognition: This module transcribes audio of spoken language to the words that were spoken, in the form of a sequence of tokens, U. We use Google Cloud API [34] for this purpose.

b) Detection: This module detects instances of certain classes of objects together with their attributes and positions in the input RGB-D image and generates the object list. In our current implementation, we assume the object detector always outputs the correct list of objects on the table in front of the robot and their positions. One can replace this module by any state-of-the-art object detector, such as Mask R-CNN [35].

c) Predicate-Argument Parsing: This module takes a sequence of tokens U as input and outputs a sequence of verb frames (there could be multiple predicates in a sentence, e.g., *pick up the ball and place it in the box*). A role can be empty (represented as a question mark "?") if it is not specified in the instruction.

Predicate-argument parsing contains three steps (shown in yellow in Fig. 3). First, as humans often use a pronoun in conveying a sequential task with multiple actions (e.g., pick up the cup and put *it* in the basket), we utilize coreference resolution [36] at the beginning to replace the pronouns with the nouns to which they refer. Next, we use a semantic role labeling (SRL) model [6] to parse the sentence into a predicate-argument structure. This detects all predicates in the sentence and produces frame-like structures for each predicate. As the role definition of the SRL model is not consistent with our verb frame definitions, we then use a rule-based mapping system to map the PropBank [37] style semantic roles output by the SRL model to our verb frame roles. This process also prunes the argument to keep only the core word (e.g., shortens *the cup* to *cup*).

d) Language Model Reasoning: This key component of LMCR formalizes the commonsense reasoning task. This module takes as input a verb frame (in which an argument is missing) and a list of sensed objects in the robot's vicinity. It outputs a complete verb frame with missing information filled in via common sense.

We formalize the commonsense reasoning task as a ranking problem. This module fills the missing information in a verb frame with detected objects to create complete candidate verb frames. The main task of the Language Model Reasoning module is to assign a score to each complete verb frame. We describe how the neural network language model is structured to achieve this goal. Once each candidate verb frame is scored, the module returns the one with the highest score.

At the core of this module is a language model whose objective is to predict the probability that a sequence of words is an appropriate natural language instruction, i.e., it aligns well with the affordances of the objects given the instruction. To model the probability distribution over a sequence, a neural network language model factors the probability according to the chain rule. Using $\{u_1, u_2, \ldots, u_N\}$ to denote the entire sentence with N tokens, the chain rule can be written as,

$$p(u_1, u_2, \ldots, u_N) = \prod_{t=1}^{N} p(u_t | u_1, u_2, \ldots, u_{t-1}), \quad (1)$$

where $p(u_t|u_1, u_2, \ldots, u_{t-1})$ is the probability of the word u_t given the previous words and is modeled by the recurrent neural network (RNN). Our method uses the language model to assign a score to a complete verb frame indicating its probability. The input frame is first converted to a word sequence using one of the linearization strategies discussed above. Then the probability of this sequence, according to the language model objective, is treated as the plausibility score.

Our neural network language model architecture consists of three layers, namely, an embedding layer, a recurrent neural network with gated recurrent units (GRU) and an output Softmax layer. The network structure is shown in Fig. 4A. First, each word in the input sequence is represented as a one-hot vector over the entire vocabulary (to train a language model, vocabulary should be determined in advance, containing all possible tokens) and fed into the embedding layer, in order to get a dense word vector representation x_t at time step t. Second, these vectors are fed to the GRU based RNN layer. The GRU is illustrated in Fig. 4B and the computation performed by GRU is formalized as follows:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]), \quad (2)$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]), \quad (3)$$
$$\tilde{h}_t = \tanh(W \cdot [r_t \circ h_{t-1}, x_t]), \quad (4)$$
$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t, \quad (5)$$

where σ is the sigmoid function, $[\cdot]$ stands for vector concatenation, $\circ$ stands for element-wise multiplication, and W_z, W_r, and W are learnable parameters, which are shared between time steps. The recurrent operation allows the network to memorize what has been seen before and make predictions based on this memory. Finally, the hidden state h_t in the t-th

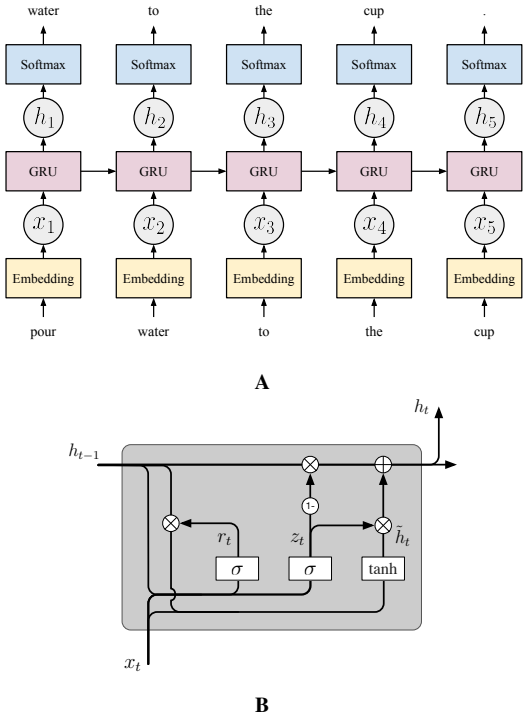


Fig. 4 LMCR's neural network architecture (**A**) The structure of LMCR's Recurrent Neural Network (RNN) with Gated Recurrent Units (GRU). (**B**) The computation of a GRU (for more details, see [38]).

time step is decoded via a learnable matrix W_o, and fed to a Softmax classifier to get the output,

$$o_t = \text{softmax}(W_o \cdot h_t). \tag{6}$$

The output o_t is a distribution over the vocabulary (the length of o_t is the same as the size of the vocabulary), representing the probability of the next word. The ground truth next word is provided by the input sentence itself. The network is trained with a negative log-likelihood objective, which compares the predicted token with the ground truth next token,

$$\mathcal{L} = -\frac{1}{N}\sum_{t=1}^{N} \hat{o}_t^{\mathrm{T}} \log o_t, \tag{7}$$

where N is the total number of time steps. We use stochastic gradient descent to train the neural network language model. As the ground truth next token is naturally provided in the input sequence, the training of the language model is unsupervised.

After the language model has been trained, when using it to evaluate a sequence we use the loss, (7), to compute the score function for the commonsense reasoning task as,

$$g_{\mathrm{LM}}(f) = \exp(-N\mathcal{L}), \tag{8}$$

which changes the loss back to the probability of the entire query sentence.

In order to use the language model to predict the correct action, we take the incomplete verb frame generated by the Predicate-Argument Parsing module, as well as the set of objects present in the scene as defined by the Detection module. We then generate a set of candidate instructions wherein the missing argument in the verb frame is filled by each of the detected objects. Since verb frames are structured representations of sentences, but the input to the LM is a sequence of words without structure, we need to convert verb frames into a sequence of words, which is called *linearization* [39], [40]. We propose two linearization methods. The first is to concatenate the predicate and all arguments directly, i.e., to use the verb, first argument, and the second argument as the input sequence. This results in unnatural sounding word sequences. The second approach is to make a more natural sentence from the frame using a rule-based approach. For example, (pour, `Theme`: water, `Destination`: cup) is converted to *pour water cup* with the former approach and *pour water to the cup* with the latter one. We refer to the LM trained and tested with the former approach as frame-based LM and the latter one as sentence-based LM. The training and testing schemata for the two variations are depicted in Fig. 5A and Fig. 5B, respectively. Although the sentence-based LM requires more complex conversion rules at test time, the training process becomes trivial by using raw sentences in the corpora, whereas the frame-based one requires a predicate-argument parsing step before training.

The generated sequences are scored using the language model. LMCR then selects the candidate verb frame with the highest score. The selected verb frame, which is complete with no missing information, is then sent to the Motion Planning module for execution by the robot.

e) Motion Planning: The motion planner takes a complete verb frame as input and computes a motion for the robot to execute the specified task. The Motion Planning module first localizes in 3D space (relative to the robot) the relevant objects corresponding to arguments in the complete verb frame (e.g., the cup corresponding to the destination of the water in the example in Fig. 1). In our implementation of the Motion Planning module, we created a task-specific motion planner for each verb (e.g., pour) that the robot should be capable of executing. For each task-specific motion planner, we defined a sequence of robot end-effector milestones based on the task specified by the verb. We represent each milestone in a coordinate system relative to a task-relevant object (e.g., for the "pour" verb, the water source is directly over the destination for the milestone corresponding to the start of pouring), which allows the robot to be robust to different placements of the task-relevant objects in the environment. We define these milestones via kinesthetic demonstrations in which we record the end effector's pose relative to the task-relevant object. When LMCR passes an complete verb frame to the Motion Planning module, the module executes the task-specific motion planner for the associated verb. The motion planner computes feasible robot motions that avoid obstacles and move the end-effector to the milestones with respect to the positions of the task-relevant objects from the instruction. We use the motion planning toolkit MoveIt! [28]

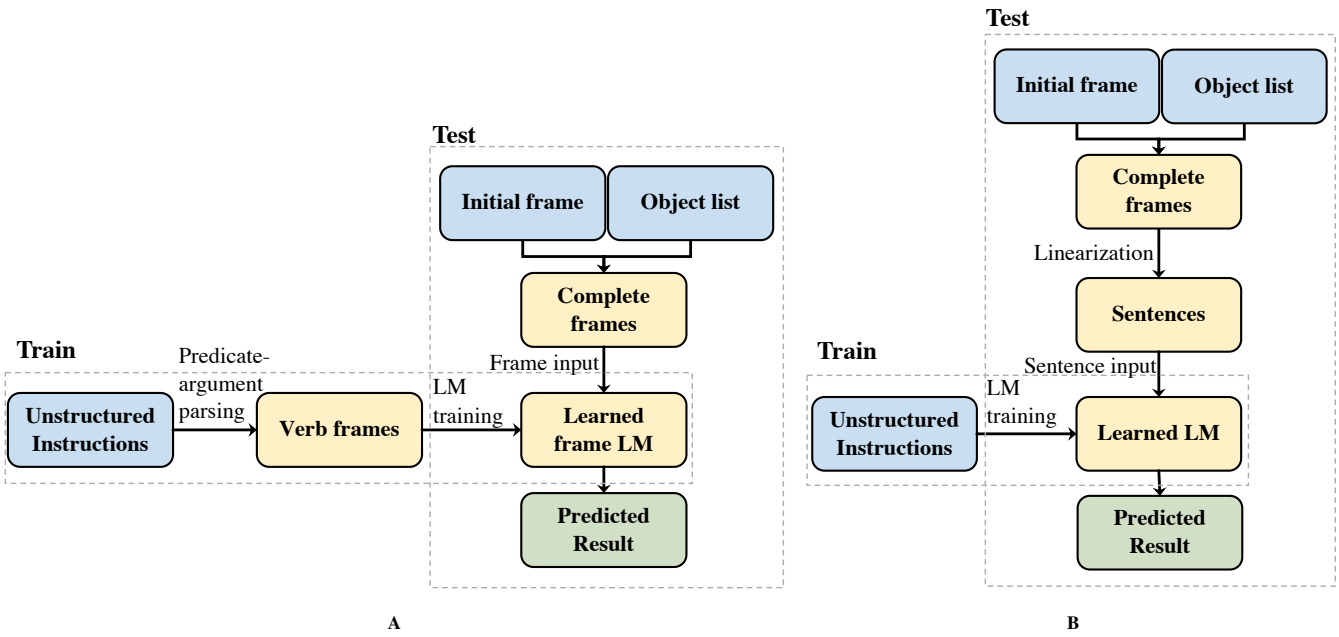


Fig. 5 Linearization strategies for the language model. (**A**) Training and testing schema for frame-based LM. (**B**) Training and testing schema for sentence-based LM.

to compute a motion plan for the robot that reaches each milestone to accomplish the task.

C. Training Corpora for Commonsense Knowledge Learning

The training data for LMCR's language model come from textual corpora, which can be treated as the knowledge source of the method. We use YouCook2 [8] and Now You're Cooking (NYC) [9] as training corpora. YouCook2 is a large instructional video dataset designed to facilitate video captioning research. The cooking steps for each video are annotated with temporal boundaries and described by imperative English sentences, resulting in around $14,000$ raw descriptions of cooking actions. NYC contains over $150,000$ recipes, each containing a step-by-step description of how to execute the recipe. We use the train, validation, and test splits provided in [41], containing about $1,400,000$ sentences (note that each recipe contains several sentences) in the training split and $10,000$ in both the validation and test splits.

These two knowledge sources have different characteristics. Sentences in YouCook2 are collected by showing annotators cooking videos and tasking them with writing descriptions for the videos. On the other hand, NYC is collected from online recipe data directly, containing some unrelated information, such as comments in the recipe unrelated to the cooking task. From a quantitative perspective, note that not all sentences in the training data are complete (i.e., do not have missing roles). The sentences that are not complete may be less helpful when training the model but are still used in the training process. We calculate a frame fulfilling rate (FFR) metric to evaluate the completeness of the instructional sentences in the two datasets, which is the percentage of complete sentences in all sentences in a corpus. The FFR values of YouCook2 and NYC are 48.95% and 30.21% respectively, implying that NYC contains a larger number of incomplete instructions, and is noisier than YouCook2.

D. Human Generated Plausibility Score Collection and Scenario Construction

In order to quantitatively evaluate LMCR's commonsense reasoning for robotic assistance instructions, we created a new human-generated dataset, since existing datasets on commonsense reasoning [29], [30], [31] are not specific to our domain of filling in missing information in instructions for robotic assistance tasks. We use the crowdsourcing platform Amazon Mechanical Turk (AMT) [42]. We provided human annotators with sentences representing completed verb frames and asked them to give a *plausibility rating* for each of them. The score scales from 1 (most implausible) to 5 (most plausible). For the example above, we would have all possible complete sentences such as "pour water to the cup", "pour water to the rope", etc. (the first of these two examples should have a higher plausibility rating). We use the resulting plausibility ratings to determine the ground truth in each scenario, which is randomly generated from the pre-specified verb and noun vocabularies. The data collection process is shown in Fig. 6.

To generate this dataset, we first manually design a set of frames with one missing role, namely the "incomplete verb frames" in Fig. 6 and their associated vocabulary $\mathcal{A}$. Then we get the set of complete verb frames by filling every word $a \in \mathcal{A}$ into the incomplete verb frames. For example, if we have a set of frames with one missing role containing two frames:

(i) (pour, `Theme: water`, `Destination: ?`),
(ii) (brush, `Theme: oil`, `Destination: ?`),

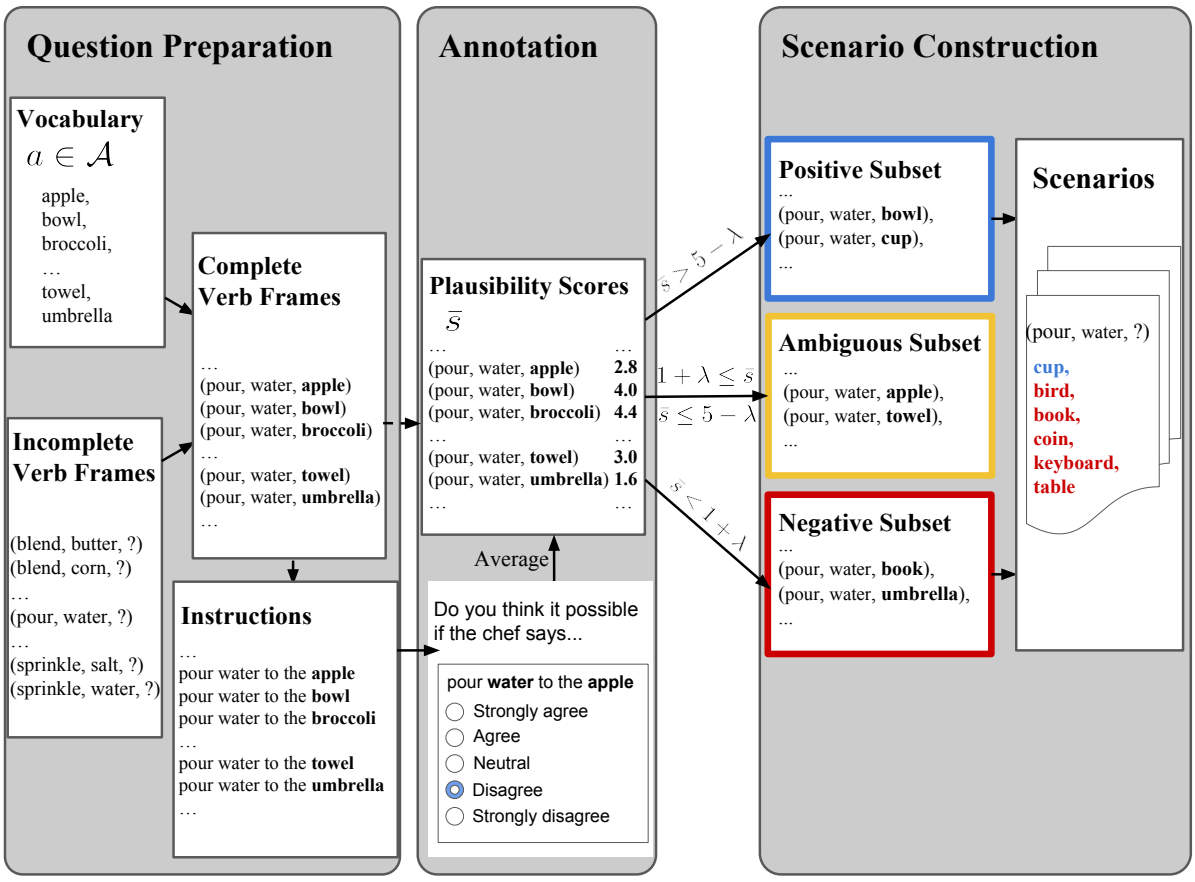


Fig. 6 Human judgment collection and scenario preparation.

and the vocabulary contains 2 words "cup" and "potato", then the resulting complete verb frame set contains,

(i) (pour, `Theme:` water, `Destination:` cup),
(ii) (pour, `Theme:` water, `Destination:` potato),
(iii) (brush, `Theme:` oil, `Destination:` cup),
(iv) (brush, `Theme:` oil, `Destination:` potato).

To make the complete verb frames more human-readable, we convert each to natural language instructions. For example, (pour, `Theme:` water, `Destination:` cup) is converted to "*pour water to the cup*". We then have human workers on Amazon Mechanical Turk (AMT) rate the plausibility of the instructions. Each complete verb frame is annotated by 5 different workers with a score from 1 (most implausible) to 5 (most plausible) and the final plausibility score is the average of 5 annotations. The plausibility score of the i-th complete verb frame is denoted as $\bar{s}_i$, and is denoted as $\bar{s}$ in Fig. 6 for simplicity.

We divide the complete verb frame set by their plausibility scores into a positive, ambiguous, and negative subset via a "plausibility threshold" λ. Specifically, those frames with score $\bar{s}_i > 5 - \lambda$ goes into the positive subset, $\bar{s}_i < 1 + \lambda$ into the negative subset, and all others into the ambiguous set. The plausibility threshold λ is an adjustable parameter and can be viewed as a measure of ambiguity. Larger threshold results in frames in the positive and negative subsets that even humans are unsure about.

Each scenario in our evaluation contains a frame with one missing role and a list of objects. Equivalently, it can also be interpreted as a list of complete verb frames $f = (v, r_1, a_1, r_2, a_2)$ with the same v, r_1, a_1, and r_2 as the initial incomplete verb frame, but different a_2's (from the list of objects). We construct a scenario by combining one frame from the positive subset and $k-1$ verb frames randomly sampled from the negative subset, keeping v, r_1, a_1, and r_2 the same. In this way, we get a set of scenarios with k objects in the list and only one positive role filler. Note that if there are multiple or no possible role filler in the list, the agent may need to ask for clarification to make the right decision, which is outside the scope of this work. The number of objects per scenario k is another adjustable parameter in our experiment. A larger k brings more negative candidates into a scenario, which makes the task more challenging.

E. Comparison Methods for Commonsense Reasoning Evaluation

As described above, LMCR gives a score to each complete verb frame $f = (v, r_1, a_1, r_2, a_2)$ in a generated list, and the highest scored one is picked as the output. We use $g_*(f)$ to denote the scoring function. In the following, we describe the scoring function of co-occurrence, Word2Vec, and ConceptNet, against which we compare our LMCR method.

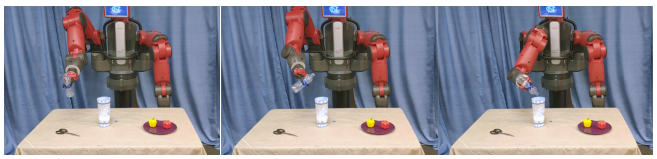


Fig. 7 An example of LMCR in action. In this example, the robot with LMCR is holding a spoon and is in front of a table with a cup, scissors, a plate, and two bell peppers. The robot receives the instruction "Could you please pour me some water?", LMCR autonomously resolves the destination of pour to "cup", and the robot then executes a motion plan to accomplish the task. For the full video, see `https://youtu.be/W5wYFd7aJP0`.

(i) **Co-occur.** The scoring function for this method, which is shorthand for "co-occurrence," is defined as

$$g_{\text{cooccur}}(f) = \text{cooccur}(v, a_2) + \text{cooccur}(a_1, a_2), \quad (9)$$

where $\text{cooccur}(x, y)$ denotes the total normalized co-occurrence score of x and y in YouCook2 and NYC sentences. This is computed by $\text{count}(x, y)/(\text{count}(x) * \text{count}(y))$ where $\text{count}(x)$ and $\text{count}(y)$ are the occurrences of x and y individually in the corpus and $\text{count}(x, y)$ is the count of x and y co-occurring in the same sentence. The normalized co-occurrence score for a pair (x, y) in YouCook2 and NYC are computed separately and added together to get the final normalized co-occurrence score $\text{cooccur}(x, y)$.

(ii) **Word2Vec.** The scoring function of the Word2Vec based method is defined as

$$g_{\text{word2vec}}(f) = -(\text{dist}(v, a_2) + \text{dist}(a_1, a_2)), \quad (10)$$

where $\text{dist}(x, y)$ denotes the Euclidean distance of word embeddings of x and y. We use GloVe embeddings [27] for this comparison.

(iii) **ConceptNet.** We use the relatedness score provided by the ConceptNet API[43], and compute the score for a frame f as

$$g_{\text{ConceptNet}}(f) = \text{rel}(v, a_2) + \text{rel}(a_1, a_2), \quad (11)$$

where $\text{rel}(x, y)$ denotes the ConceptNet relatedness score [44] of x and y.

III. Results

We implemented LMCR on a Baxter robot [45] and demonstrate the robot's ability to listen to an (incomplete) instruction spoken by a human user, correctly infer the action to perform by filling in the missing information, and execute the action in the physical world. We also use the collected data set of human judgments to quantitatively demonstrate the ability of LMCR to correctly fill in missing information in incomplete natural language instructions.

A. LMCR in Action

We deploy LMCR on a Baxter robot [45], a research robotics platform with two arms, each with 7 degrees of freedom, and demonstrate its ability to successfully accomplish intended tasks given incomplete spoken instructions in different scenarios. We show an example in Fig. 7. In these scenarios, LMCR first successfully parses the spoken language instruction to their corresponding incomplete verb frames. Given the set of objects in front of the robot, LMCR then fills in the missing role with the most likely object. Finally, the robot plans a motion based on the verb in the instruction and the location of the relevant objects and successfully completes the intended task.

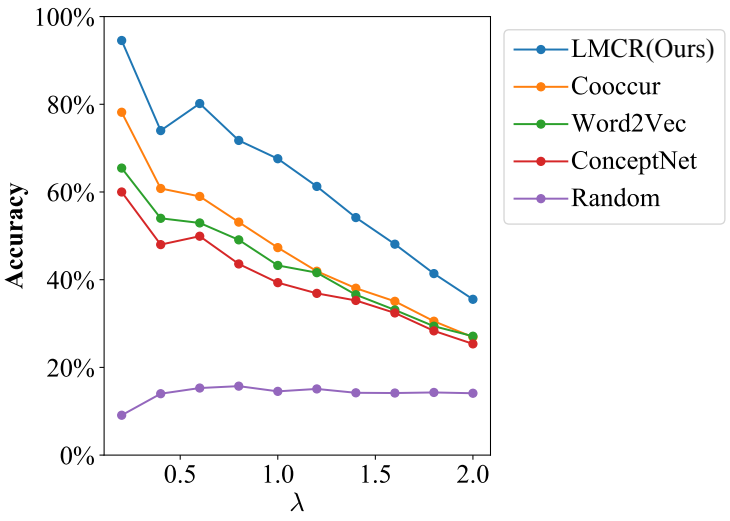
B. Accuracy of Filling Missing Information Via Common Sense

We quantitatively evaluate the accuracy of LMCR as it fills in missing information in incomplete instructions using the collected human judgment dataset. In our evaluation, we consider 11 verbs listed in the leftmost column of Fig. 8C. We change the plausibility threshold λ and the number of object in the list k to create scenarios with various difficulties. We compare our method LMCR against other methods described in the above section, namely, Random, Co-occur, Word2Vec, and ConceptNet.

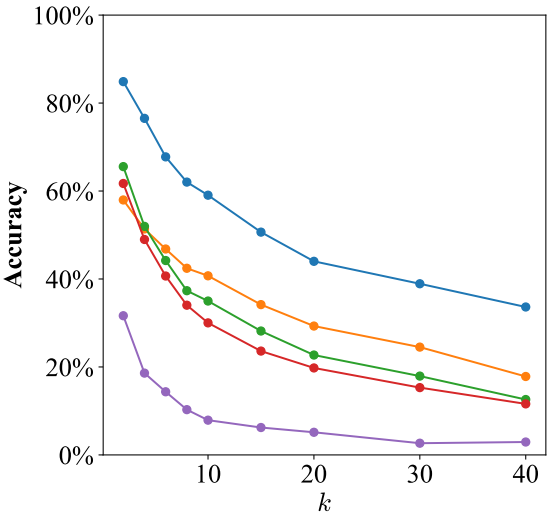
We evaluate each method by calculating its accuracy, which is the percent of successful predictions over all scenarios. The results are shown in Fig. 8. Our method LMCR outperforms, overall, each of the competing methods, under variations in λ (the threshold splitting the positive and negative subsets) and in k (the number of objects the method is choosing from in each scenario). LMCR performs consistently better than other methods when considering all actions, for all variations of λ and k, as shown in Fig. 8A and Fig. 8B respectively. In Fig. 8C, where we fix $\lambda = 1.0$ and $k = 6$ and show the accuracy on single actions, there are certain actions for which other methods outperform our method, specifically "dump", "fry", and "heat". However, our method outperforms the others when considering all actions. This suggests that, overall, LMCR better encodes they type of commonsense reasoning we are addressing in this work. Also, as λ grows (see Fig. 8A), the performance of all methods decrease. This is because a larger λ will introduce more challenging object candidates which are closer to the "neutral" plausibility rating of 3.0. Also, as k grows (see Fig. 8B), more objects are added to the object list for each scenario. This results in a performance decrease for all methods. Intuitively, increasing k will make the scenario more cluttered and difficult. Our results show that LMCR can successfully learn commonsense knowledge from textual corpora, and results in the ability to predict objects that align well with humans in the home assistance domain.

C. Comparison of Training Methods for LMCR's Language Model

In this set of experiments, we compare several different ways of training the language model (LM) used by the Language Model Reasoning module of LMCR. We compare LM trained with two linearization strategies, namely, frame-based and sentence-based. We also compare different training corpora, by training LM with YouCook2 data only, with Now



A



B

	Random	Cooccur	Word2Vec	ConceptNet	LMCR (Ours)
Blend	14.6%	47.9%	63.7%	59.8%	**77.1%**
Brush	14.1%	45.6%	45.3%	26.3%	**76.8%**
Dip	14.7%	26.1%	32.2%	40.0%	**42.6%**
Dump	14.6%	**60.7%**	44.3%	42.9%	55.7%
Fill	14.5%	46.6%	21.6%	27.5%	**86.9%**
Fry	14.3%	**82.9%**	45.7%	62.9%	54.3%
Heat	13.8%	50.0%	17.5%	**52.5%**	37.5%
Pour	14.8%	46.2%	49.3%	45.9%	**54.5%**
Rub	14.3%	44.5%	41.0%	30.5%	**68.5%**
Season	14.2%	40.0%	26.0%	58.0%	**74.0%**
Sprinkle	14.6%	57.9%	52.6%	33.7%	**77.5%**
Total Accuracy	14.4%	48.3%	45.1%	40.8%	**70.5%**

C

Fig. 8 Accuracy results for LMCR. We evaluate the ability of LMCR to correctly fill in missing information for scenarios of varying difficulty. Threshold λ and number of objects per scenario k are used to control the "difficulty" of commonsense reasoning task. A larger threshold λ allows for more "ambiguous" verb frames (verb frame with human annotated plausibility score near 3.0) to be included in the evaluation, and a larger number of objects per scenario k brings more incorrect objects into each scenario, both making the task more difficult. (**A**) Overall accuracy with λ changing from 0.2 to 2.0 and $k = 6$ for all methods. The number of data points will also change with λ, which is also depicted in the figure. (**B**) Overall accuracy with k changing from 2 to 40 and $\lambda = 1.0$. (**C**) Overall accuracy and accuracy per predicate for $\lambda = 1.0$ and $k = 6$.

You're Cooking (NYC) data only and the combination of the two.

Results are shown in Fig. 9A and Fig. 9B which vary λ and k respectively. As shown in the result, sentence-based LM generally performs better than frame-based LM. We suspect this is due to the fact that the former is end-to-end trained while the latter requires generated training data from an upstream predicate-argument parser, whose errors may propagate to the training process of the frame-based LM. Also, the parser cannot generate a frame for predicates that are not in its verb vocabulary, even though these relatively rare predicates can be beneficial when learning others. For example, "scatter some salt on the beef" would help with the learning for "spread" and "sprinkle" as they can be synonyms. While the sentence-based LM can take advantage of this, the frame-based schema loses this information in the training data, since "scatter" is not among the 11 verbs we consider. For sentence-based LM, the performance of the models trained with NYC and YouCook2+NYC data are similar, and both are better than the model trained on YouCook2 alone. For frame-based LM, the mixed data of the two yields the best performance. Although NYC is noisier than YouCook2, the former still brings positive input to the language model training, since its scale is much larger than the latter. This suggests that the language model is robust to the noise in the training data. These results demonstrate that a human annotated dataset, such as YouCook2, is not necessarily better than a recipe-based dataset, although it

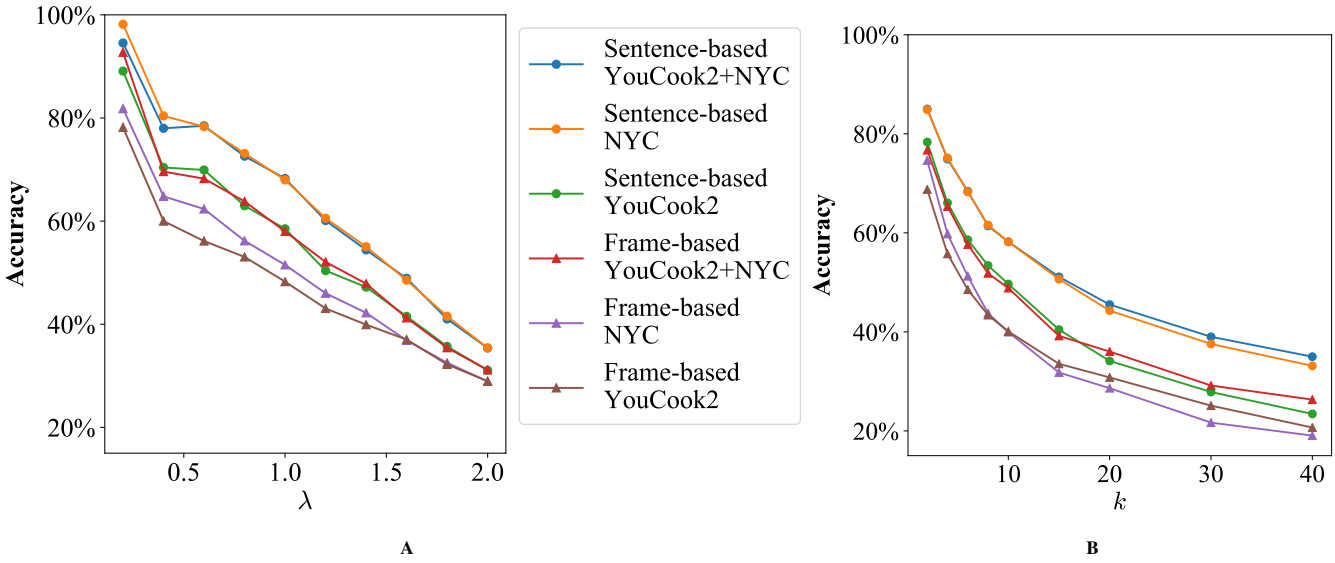


Fig. 9 Result of different linearization strategies and datasets. (**A**) Overall accuracy of all methods with λ changing from 0.2 to 2.0 and $k = 6$. (**B**) Overall accuracy of all methods with k changing from 2 to 40 and $\lambda = 1.0$.

may still be helpful. Based on the above analyses, we use the sentence-based LM trained on the mixed data of YouCook2 and NYC when comparing with other commonsense reasoning approaches in the previous section.

IV. Discussion

We demonstrated that a robot with LMCR is able to execute tasks specified by natural language instructions and resolve missing information in the instructions. Here we place our work in the context of related work in robotics, natural language processing (NLP), psycholinguistics, and information retrieval (IR), and discuss limitations and improvements for future work.

A. Language Grounding And Structured Representation of Instructions

To understand natural language instructions, a robot has to extract a semantically meaningful representation of natural language and ground it to the perceptual elements and actions in its environment. This process is referred to as language grounding. Several approaches have been proposed for language grounding, which can be broadly divided into probabilistic models [10], [11], [12], [13], [14], [15] and deterministic models [46], [47], [48], [16], [17], [49]. These approaches seek to find an intermediate representation in order to bridge natural language and machine commands. Our proposed model falls into the deterministic model category. However, to the best of our best knowledge, grounding unstated concepts with commonsense knowledge has not been considered in previous literature. Recently, due to the advancement of deep neural networks, several works use sequence learning and reinforcement learning to directly map text to actions, skipping the need of an intermediate representation of instructions [50], [51], [52], [53], [54], [55], [56], [57]. However, they either consider only navigation tasks, or a simple game environment, where the possible actions are limited. In contrast, our method generalizes to any task domain as we can easily extend the set of our verb frame representations by adding more frames to our training corpus.

One issue with the verb frame structure is that there is no universal agreement on how a frame should be defined. Existing corpora such as FrameNet [58] and VerbNet [59] define a global thematic role set, i.e., the meanings of roles are consistent across frames, while PropBank [37] defines roles for each verb. There are mappings between the three resources in the Unified Verb Index [60]. We base our verb frame definitions on VerbNet.

B. Semantic Affordance and its Relevance to LMCR

Affordance can be defined as knowledge of an object's functionality. Understanding affordances is crucial for a robotic system to recognize human activities, interact with the environment, and achieve its goal [29]. Previous research on affordance can be divided into two categories, namely, visual affordance and semantic affordance. In visual affordance, the problem of *affordance segmentation* has long been studied by the computer vision community [61], [62], [63]. Affordance segmentation aims to find the functional part of a tool in an image. Semantic affordance considers the concept in a different way. For example, Zhu et al. [64] represent the affordance of an object via an affordance label (e.g., edible), a human pose representation of the action, and a relative position of the object with respect to the human pose (e.g., next to). Chao et al. [29] looked into mining semantic affordance, i.e., inferring whether a given action can be performed on an object (e.g., pour water versus pour table). They propose to mine semantic affordance knowledge

from textual and visual data but did not consider a learning-based approach as our neural network language model does.

Psychologists view affordance as an important aspect of how humans perceive the world. Gibson [65] suggests humans perceive objects by looking at their affordances. A recent study [66] has shown that perceived affordance is at least equally as important as an object's appearance for humans when recognizing objects. Though it is still hard for a robot to have human-level knowledge and recognition ability, an agent with affordance knowledge has the potential to interact with humans and its environment more efficiently. Our work extends the previous definition of semantic affordance [29] to predicting not only the direct object of a predicate but also every role in its corresponding verb frame, depending on both the predicate and other roles. This is an important step toward a robot achieving human-level object recognition and scene understanding.

C. Selectional Preference And Thematic Fit Modeling

Another research topic closely related to this work is selectional preference [67], [68], [69], [70], [71], [72], [73], [23], [24], [25] or similarly, thematic fit modeling [18], [19], [20], [21], [22]. The task of thematic fit modeling is to determine how likely a noun can be filled into a role associated with a predicate [74] (e.g., how likely is *cup* to be the `Destination` of *pour*). The models are often evaluated by comparing plausibility ratings produced by the models compared to human-generated data [75].

Computational models for thematic fit are largely based on the distributional hypothesis [76], [77], which can be stated as the degree of semantic similarity between two linguistic expressions A and B is a function of the similarity of the linguistic contexts in which A and B can appear [78]. Based on this assumption, one popular approach is to build a single prototype vector for each role by averaging the dependency-based vectors of its most typical fillers [79], [20], [80]. At the testing phase, the distance between the word embedding of the candidate argument and the prototype vector is computed as the signal of thematic fitness. Some work adds count-based metrics as a measure of typicality to get more relevant prototype vectors [75]. However, these approaches usually assume the accessibility of structured representations of sentences such as dependency trees or semantic role labels, which are generated either from a corpus with pre-annotated structure information or a pre-trained semantic parser. To the best of our knowledge, none of these methods have tried to train a language model on natural sentences directly for thematic fit modeling. In the real world, structured information is not always available, especially in a specialized target domain, such as robotic home assistance. Even if we can use a pre-trained semantic parser, we cannot expect the parser to be perfect and thus will get incomplete, or even incorrect information during training. By contrast, our proposed neural network language model requires only plain text to train.

D. Knowledge Graphs vs. Neural Networks for Knowledge Representation

Knowledge graph representations have been widely used in robotic systems representing world knowledge. For example, Liu et al. [81] propose the use of Attributed Relational Graphs to represent concepts in human-robot dialogue and the perceived environment in a single place during human-robot interaction. Outside the robotics community, knowledge graph methods are also widely used for representing commonsense knowledge [82], [64], [83]. Generally speaking, however, the construction of such graphs requires hand-designing graph structures and information retrieval methods. This requires substantial human effort, may result in bias due to pre-defined relationships, and may lack the ability to generalize.

LMCR, by contrast, uses a neural network language model and is based on the intuition that world knowledge is implicitly encoded in textual corpora. For example, if the phrase "*pour some oil into the pan*" occurs frequently in the corpora, this implies that oil is something that can be poured and that pan is something into which oil can be poured. By learning statistics on the language corpora, we are implicitly learning these physical relationships, which we leverage to resolve incomplete instructions. The end-to-end learning nature of the neural network language model makes it free of cumbersome knowledge graph designs and allows for better generalization.

In the natural language processing (NLP) community, neural network language models have been applied successfully to improve various downstream applications [84], [85], including commonsense reasoning. For example, Trinh et al. [86] applied pre-trained language models to solve the Winograd Schema Challenge, a pronoun disambiguation challenge which involves using common sense to determine the object pronouns are referring to (e.g., the trophy does not fit in the suitcase because *it* is too big). Recently, a large transformer-based language model BERT [87] has achieved state-of-the-art results on several NLP benchmarks, including a commonsense reasoning task Situations With Adversarial Generations (SWAG) [88], by pre-training the language model on the unlabeled textual corpus and then fine-tuning on a specific task. These applications show that neural network language models are well suited to encoding and extracting knowledge that exists in large language corpora. In our current work, we have shown that our language model is able to learn domain-specific knowledge by using raw textual data from the web and further that the learned knowledge matches well with human common sense.

E. Limitations and Future Work

In this work, we restrict each scenario to have an incomplete verb frame with only one missing role. While an important step toward generalized common sense, in future work we plan to consider an arbitrary number of missing roles. We are also currently investigating the integration of recent, larger, masked language models, such as BERT [87], into our framework. Finally, we intend to apply the

same commonsense reasoning approach to other domains in addition to home assistance to enable the system to understand a broader range of instructions.

ACKNOWLEDGEMENT

We thank Y. Nie and L. Yu for helpful suggestions and D. Gao, X. Lu, M. Fu and C. Bowen for helping with data collection. This research was supported in part by the U.S. National Science Foundation (NSF) under Awards IIS-1149965 and CCF-1533844, ARO-YIP Award #W911NF-18-1-0336, and a Google Faculty Research Award. The views contained in this article are those of the authors and not of the funding agency.

REFERENCES

[1] D. Jurafsky and J. H. Martin, *Speech and language processing*, vol. 3. Pearson London, 2014.

[2] X. Carreras and L. Màrquez, Introduction to the conll-2005 shared task: Semantic role labeling, in *Proceedings of the ninth conference on computational natural language learning*, pp. 152–164, Association for Computational Linguistics, 2005.

[3] V. Punyakanok, D. Roth, and W.-t. Yih, The importance of syntactic parsing and inference in semantic role labeling, *Computational Linguistics*, vol. 34, no. 2, pp. 257–287, 2008.

[4] N. FitzGerald, O. Täckström, K. Ganchev, and D. Das, Semantic role labeling with neural network factors, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 960–970, 2015.

[5] D. Marcheggiani, A. Frolov, and I. Titov, A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling, *arXiv preprint arXiv:1701.02593*, 2017.

[6] L. He, K. Lee, M. Lewis, and L. Zettlemoyer, Deep semantic role labeling: What works and whats next, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 473–483, 2017.

[7] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, Recurrent neural network based language model, in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[8] L. Zhou, C. Xu, and J. J. Corso, Towards automatic learning of procedures from web instructional videos, in *AAAI Conference on Artificial Intelligence*, 2018.

[9] "Now you're cooking recipe dataset." `http://www.ffts.com/recipes.htm`, 2013. [Online; accessed 09-October-2018].

[10] S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. Teller, and N. Roy, Understanding natural language commands for robotic navigation and mobile manipulation, in *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pp. 1507–1514, AAAI Press, 2011.

[11] T. M. Howard, S. Tellex, and N. Roy, A natural language planner interface for mobile manipulators, in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6652–6659, IEEE, 2014.

[12] F. Duvallet, M. R. Walter, T. Howard, S. Hemachandra, J. Oh, S. Teller, N. Roy, and A. Stentz, Inferring maps and behaviors from natural language instructions, in *Experimental Robotics*, pp. 373–388, Springer, 2016.

[13] S. Hemachandra, F. Duvallet, T. M. Howard, N. Roy, A. Stentz, and M. R. Walter, Learning models for following natural language directions in unknown environments, in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5608–5615, IEEE, 2015.

[14] R. Paul, J. Arkin, N. Roy, and T. Howard, Grounding abstract spatial concepts for language interaction with robots, in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 4929–4933, AAAI Press, 2017.

[15] R. Paul, J. Arkin, D. Aksaray, N. Roy, and T. M. Howard, Efficient grounding of abstract spatial concepts for natural language interaction with robot platforms, *The International Journal of Robotics Research*, p. 0278364918777627, 2018.

[16] D. K. Misra, J. Sung, K. Lee, and A. Saxena, Tell me dave: Context-sensitive grounding of natural language to manipulation instructions, *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 281–300, 2016.

[17] D. K. Misra, K. Tao, P. Liang, and A. Saxena, Environment-driven lexicon induction for high-level instructions, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, pp. 992–1002, 2015.

[18] K. McRae, M. J. Spivey-Knowlton, and M. K. Tanenhaus, Modeling the influence of thematic fit (and other constraints) in on-line sentence comprehension, *Journal of Memory and Language*, vol. 38, no. 3, pp. 283–312, 1998.

[19] B. Vandekerckhove, D. Sandra, and W. Daelemans, A robust and extensible exemplar-based model of thematic fit, in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 826–834, Association for Computational Linguistics, 2009.

[20] A. Sayeed and V. Demberg, Combining unsupervised syntactic and semantic models of thematic fit, in *Proceedings of the first Italian Conference on Computational Linguistics (CLiC-it 2014)*, 2014.

[21] C. Greenberg, A. Sayeed, and V. Demberg, Improving unsupervised vector-space thematic fit evaluation via role-filler prototype clustering, in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 21–31, 2015.

[22] E. Santus, E. Chersoni, A. Lenci, and P. Blache, Measuring thematic fit with distributional feature overlap, in *Proceedings of the Conference on Empirical Methods for Natural Language Processing 2017*, 2017.

[23] T. Van de Cruys, A neural network approach to selectional preference acquisition, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 26–35, 2014.

[24] E. Shutova, N. Tandon, and G. De Melo, Perceptually grounded selectional preferences, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, pp. 950–960, 2015.

[25] M. Nadejde, A. Birch, and P. Koehn, Modeling selectional preferences of verbs and nouns in string-to-tree machine translation, in *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, vol. 1, pp. 32–42, 2016.

[26] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, Distributed representations of words and phrases and their compositionality, in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[27] J. Pennington, R. Socher, and C. Manning, Glove: Global vectors for word representation, in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[28] "Moveit!." `https://moveit.ros.org/`.

[29] Y.-W. Chao, Z. Wang, R. Mihalcea, and J. Deng, Mining semantic affordances of visual object categories, in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4259–4267, IEEE, 2015.

[30] T. Warren, E. Milburn, N. D. Patson, and M. W. Dickey, Comprehending the impossible: what role do selectional restriction violations play?, *Language, cognition and neuroscience*, vol. 30, no. 8, pp. 932–939, 2015.

[31] L. Pylkkänen and B. McElree, An MEG study of silent meaning, *Journal of cognitive neuroscience*, vol. 19, no. 11, pp. 1905–1921, 2007.

[32] D. Das, D. Chen, A. F. Martins, N. Schneider, and N. A. Smith, Frame-semantic parsing, *Computational linguistics*, vol. 40, no. 1, pp. 9–56, 2014.

[33] K. M. Hermann, D. Das, J. Weston, and K. Ganchev, Semantic frame identification with distributed word representations, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 1448–1458, 2014.

[34] "Google cloud speech-to-text." `https://cloud.google.com/speech-to-text/`.

[35] K. He, G. Gkioxari, P. Dollár, and R. Girshick, Mask R-CNN, in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, IEEE, 2017.

[36] "Neuralcoref: Coreference resolution in spacy with neural networks.." `https://github.com/huggingface/neuralcoref`, 2013.

[37] M. Palmer, D. Gildea, and P. Kingsbury, The proposition bank: An annotated corpus of semantic roles, *Computational linguistics*, vol. 31, no. 1, pp. 71–106, 2005.

[38] C. Olah, "Understanding LSTM networks, 2015." `http://colah.github.io/posts/2015-08-Understanding-LSTMs`, 2015.

[39] K. Filippova and M. Strube, Tree linearization in English: Improving language model based approaches, in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pp. 225–228, Association for Computational Linguistics, 2009.

[40] I. Konstas, S. Iyer, M. Yatskar, Y. Choi, and L. Zettlemoyer, Neural AMR: Sequence-to-sequence models for parsing and generation, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 146–157, 2017.

[41] C. Kiddon, L. Zettlemoyer, and Y. Choi, Globally coherent text generation with neural checklist models, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 329–339, 2016.

[42] "Amazon Mechanical Turk." `https://www.mturk.com/`.

[43] "Conceptnet5 API." `https://github.com/commonsense/conceptnet5/wiki/API`.

[44] R. Speer, J. Chin, and C. Havasi, Conceptnet 5.5: An open multilingual graph of general knowledge, in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[45] R. Robotics, "Baxter research robot." `www.rethinkrobotics.com/baxter-research-robot/`, 2013.

[46] L. Zettlemoyer and M. Collins, Online learning of relaxed ccg grammars for parsing to logical form, in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[47] L. S. Zettlemoyer and M. Collins, Learning to map sentences to logical form: structured classification with probabilistic categorial grammars, in *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, pp. 658–666, AUAI Press, 2005.

[48] B. J. Thomas and O. C. Jenkins, Roboframenet: Verb-centric semantics for actions in robot middleware, in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 4750–4755, IEEE, 2012.

[49] J. Thomason, A. Padmakumar, J. Sinapov, N. Walker, Y. Jiang, H. Yedidsion, J. Hart, P. Stone, and R. J. Mooney, Improving grounded natural language understanding through human-robot dialog, *arXiv preprint arXiv:1903.00122*, 2019.

[50] S. R. Branavan, H. Chen, L. S. Zettlemoyer, and R. Barzilay, Reinforcement learning for mapping instructions to actions, in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pp. 82–90, Association for Computational Linguistics, 2009.

[51] S. Branavan, L. S. Zettlemoyer, and R. Barzilay, Reading between the lines: Learning to map high-level instructions to commands, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 1268–1277, Association for Computational Linguistics, 2010.

[52] A. Vogel and D. Jurafsky, Learning to follow navigational directions, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 806–814, Association for Computational Linguistics, 2010.

[53] D. Misra, J. Langford, and Y. Artzi, Mapping instructions and visual observations to actions with reinforcement learning, in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1004–1015, 2017.

[54] A. Sinha, M. Sarkar, B. Krishnamurthy, *et al.*, Attention based natural language grounding by navigating virtual environment, *arXiv preprint arXiv:1804.08454*, 2018.

[55] M. Janner, K. Narasimhan, and R. Barzilay, Representation learning for grounded spatial reasoning, *Transactions of the Association of Computational Linguistics*, vol. 6, pp. 49–61, 2018.

[56] P. Shah, M. Fiser, A. Faust, J. C. Kew, and D. Hakkani-Tur, Follownet: Robot navigation by following natural language directions with deep reinforcement learning, *arXiv preprint arXiv:1805.06150*, 2018.

[57] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Y. Wang, and L. Zhang, Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation, *arXiv preprint arXiv:1811.10092*, 2018.

[58] J. Ruppenhofer, M. Ellsworth, M. R. Petruck, and C. R. Johnson, Framenet ii: Extended theory and practice, *http://framenet.icsi.berkeley.edu/*, 2006.

[59] K. K. Schuler, Verbnet: A broad-coverage, comprehensive verb lexicon, *Ph. D. Thesis, University of Pennsylvania*, 2005.

[60] U. of Colorado, "Unified verb index." `http://verbs.colorado.edu/verb-index/`, 2013.

[61] B. Yao, J. Ma, and L. Fei-Fei, Discovering object functionality, in *2013 IEEE International Conference on Computer Vision (ICCV)*, pp. 2512–2519, IEEE, 2013.

[62] J. Sawatzky, A. Srikantha, and J. Gall, Weakly supervised affordance detection, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5197–5206, IEEE, 2017.

[63] T.-T. Do, A. Nguyen, and I. Reid, Affordancenet: An end-to-end deep learning approach for object affordance detection, in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–5, IEEE, 2018.

[64] Y. Zhu, A. Fathi, and L. Fei-Fei, Reasoning about object affordances in a knowledge base representation, in *European conference on computer vision*, pp. 408–424, Springer, 2014.

[65] E. J. Gibson, The concept of affordances in development: The renascence of functionalism, in *The concept of development: The Minnesota symposia on child psychology*, vol. 15, pp. 55–81, Lawrence Erlbaum Hillsdale, NJ, 1982.

[66] L. M. Oakes and K. L. Madole, Function revisited: How infants construe functional features in their representation of objects, in *Advances in child development and behavior*, vol. 36, pp. 135–185, Elsevier, 2008.

[67] P. Resnik, Selectional constraints: An information-theoretic model and its computational realization, *Cognition*, vol. 61, no. 1-2, pp. 127–159, 1996.

[68] D. McCarthy and J. Carroll, Disambiguating nouns, verbs, and adjectives using automatically acquired selectional preferences, *Computational Linguistics*, vol. 29, no. 4, pp. 639–654, 2003.

[69] K. Erk, A simple, similarity-based model for selectional preferences, in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 216–223, 2007.

[70] K. Erk, S. Padó, and U. Padó, A flexible, corpus-driven model of regular and inverse selectional preferences, *Computational Linguistics*, vol. 36, no. 4, pp. 723–763, 2010.

[71] T. Van de Cruys, A non-negative tensor factorization model for selectional preference induction, *Natural Language Engineering*, vol. 16, no. 4, pp. 417–437, 2010.

[72] A. Ritter, O. Etzioni, *et al.*, A latent dirichlet allocation method for selectional preferences, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 424–434, Association for Computational Linguistics, 2010.

[73] D. O. Séaghdha and A. Korhonen, Modelling selectional preferences in a lexical hierarchy, in *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pp. 170–179, Association for Computational Linguistics, 2012.

[74] A. Sayeed, C. Greenberg, and V. Demberg, Thematic fit evaluation: an aspect of selectional preferences, in *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pp. 99–105, 2016.

[75] E. Chersoni, A. T. Urrutia, P. Blache, and A. Lenci, Modeling violations of selectional restrictions with distributional semantics, in *Proceedings of the Workshop on Linguistic Complexity and Natural Language Processing*, pp. 20–29, 2018.

[76] Z. S. Harris, Distributional structure, *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[77] M. Sahlgren, The distributional hypothesis, *Italian Journal of Disability Studies*, vol. 20, pp. 33–53, 2008.

[78] A. Lenci, Distributional semantics in linguistic and cognitive research, *Italian journal of linguistics*, vol. 20, no. 1, pp. 1–32, 2008.

[79] M. Baroni and A. Lenci, Distributional memory: A general framework for corpus-based semantics, *Computational Linguistics*, vol. 36, no. 4, pp. 673–721, 2010.

[80] A. Sayeed, V. Demberg, and P. Shkadzko, An exploration of semantic features in an unsupervised thematic fit evaluation framework, *Italian Journal of Computational Linguistics*, vol. 1, no. 1, 2015.

[81] C. Liu and J. Y. Chai, Learning to mediate perceptual differences in situated human-robot dialogue, in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[82] P. Schüller, Tackling winograd schemas by formalizing relevance theory in knowledge graphs, in *Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2014.

[83] R. Vedantam, X. Lin, T. Batra, C. Lawrence Zitnick, and D. Parikh, Learning common sense through visual abstraction, in *Proceedings of the IEEE international conference on computer vision*, pp. 2542–2550, 2015.

[84] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, Improving language understanding by generative pre-training, *URL* `https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf`, 2018.

[85] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, Deep contextualized word representations, in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL: HLT)*, (New Orleans, Louisiana, USA), 2018.

[86] T. H. Trinh and Q. V. Le, A simple method for commonsense reasoning, *arXiv preprint arXiv:1806.02847*, 2018.

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

[88] R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi, Swag: A large-scale adversarial dataset for grounded commonsense inference, *arXiv preprint arXiv:1808.05326*, 2018.