Learning from Implicit Information in Natural Language Instructions for Robotic Manipulations

Ozan Arkan Can[*†]
Koç University

Pedro Zuidberg Dos Martires[*‡]
KU Leuven

Andreas Persson
Örebro University

Julian Gaal
Osnabrück University

Amy Loutfi
Örebro University

Luc De Raedt
KU Leuven

Deniz Yuret
Koç University

Alessandro Saffiotti
Örebro University

Abstract

Human-robot interaction often occurs in the form of instructions given from a human to a robot. For a robot to successfully follow instructions, a common representation of the world and objects in it should be shared between humans and the robot so that the instructions can be grounded. Achieving this representation can be done via learning, where both the world representation and the language grounding are learned simultaneously. However, in robotics this can be a difficult task due to the cost and scarcity of data. In this paper, we tackle the problem by separately learning the world representation of the robot and the language grounding. While this approach can address the challenges in getting sufficient data, it may give rise to inconsistencies between both learned components. Therefore, we further propose Bayesian learning to resolve such inconsistencies between the natural language grounding and a robot's world representation by exploiting spatio-relational information that is implicitly present in instructions given by a human. Moreover, we demonstrate the feasibility of our approach on a scenario involving a robotic arm in the physical world.

1 Introduction

Consider yourself standing in your kitchen and having your robot assist you in preparing tonight's meal. You then give it the instruction: *'fetch the bowl next to the bread knife!'*. For the robot to correctly perform your intended instruction, which is grounded in your world representation, it must correctly ground your natural language instruction into its own world representation.

This small scenario already introduces the two key components of language grounding in robotics: the construction of a world representation from sensor data and the grounding of natural language into the constructed representation. Ideally these two components would be learned in a joint fashion (Hu et al., 2017a; Johnson et al., 2017; Santoro et al., 2017; Hudson and Manning, 2018; Perez et al., 2018). However, the scarcity of data makes this approach impractical. The millions of data points necessary for state-of-the-art joint computer vision and natural language processing are simply non-existing. We opt, therefore, to separately learn the world representation component and the language grounding component.

One approach for constructing a world representation of a robot is through so-called *perceptual anchoring*. Perceptual anchoring handles the problem of creating and maintaining, over time, the correspondence between symbols in a constructed world model and perceptual data that refer to the same physical object (Coradeschi and Saffiotti, 2000). In this work, we use sensor driven bottom-up anchoring (Loutfi et al., 2005), whereby anchors (symbolic representations of objects) can be created by perceptual observations derived directly from the input sensory data. When modeling a scene, based on visual sensor data, through object anchoring, noise and uncertainties will inevitably be present. This leads, for example, to a green 'apple' object being incorrectly anchored as a 'pear'.

For the language grounding, we opt to perform the learning on **synthetic data** that simulates the world represented as anchors. This means that we do not ground the language using sensor data as

[*]Equal contribution
[†]ocan13@ku.edu.tr
[‡]pedro.zuidbergdosmartires@cs.kuleuven.be

signal but a symbolic representation of the world. During training these symbols are synthetic and simulated, and during the deployment of the language grounding these are anchors provided by an anchoring system. As the real world is inherently relational and as natural language instructions are often given in terms of spatial relations as well, the learned language grounder must also be able to ground spatial language such as *'next to'*.

As a result of learning the construction of a world model and the language grounding separately, **contradictions** arise **between** the world **representations** of a human and a robot. The supervision that an instruction would give to a robot is not present when learning the representation of the world of a robot. These inconsistencies then propagate through to inconsistencies between the instructions a human gives to a robot and the robot's world model. To ensure that a robot is able to correctly carry out an instruction, such inconsistencies must be resolved and the world model of the robot be matched to the world model of the human.

This is not the first paper that tackles the problem of belief revision in robotics. However, prior work (Tellex et al., 2013; Thomason et al., 2015; She and Chai, 2017), with the notable exception of (Mast et al., 2016), relied on explicit information transfer between humans and robots when inconsistencies arose in grounded language and the robot's world representation. An example would be a robot asking clarification questions until it is clear what the human meant (Tellex et al., 2013).

We propose an approach that probabilistically reasons over the grounding of an instruction and a robot's world representation in order to perform Bayesian learning to update the world representation given the grounding. This is closely related to the work of Mast et al. who also deploy a Bayesian learning approach. The key difference, however, is that they do not learn the language component but ground a description of a scene by relying on a predefined model to ground language. We demonstrate the validity of our approach for reconciling instructions and world representations on a showcase scenario involving a camera, a robot arm and a natural language interface.

2 Preliminaries

The overarching objective of our system is to plan and execute robot manipulation actions based on natural language instructions. Presumptuously, this requires, in the first place, that both the planner of the robot manipulator, as well as the natural language grounder (cf. Section 2.2), share a joint semantically rich object-centered model of the perceived environment, i.e., a *semantic world model* (Elfring et al., 2013).

2.1 Visual Object Anchoring

In order to model a semantic object-centered representation of the external environment, we rely upon the notions and definitions found within the concept of perceptual anchoring (Coradeschi and Saffiotti, 2000). Following the approach for sensor-driven bottom-up acquisition of perceptual data, as described by (Persson et al., 2019), the used anchoring procedure is, initially, triggered by sensory input data provided by a *Kinect2 RGB-D sensor*. Each frame of input *RGB-D* data is, subsequently, processed by a *perceptual system*, which exploits both the visual 2-D information, as well as the 3-D depth information, in order to: *1)* detect and segment the subset of data (referred to as *percepts*), that originates from a single individual object in the physical world, and *2)* measure *attribute values* for each segmented percept, e.g., measuring a *position attribute* as the $\mathbb{R}^3$ geometrical center of an object, or a visual *color attribute* measured as a color histogram (in *HSV* color space).

The percept-symbol correspondence is, thereafter, established by a *symbolic system*, which handles the grounding of measured attributes values to corresponding predicate symbols through the use of *predicate grounding relations*, e.g., a certain peek in a color histogram, measured as a *color attribute*, is mapped to a corresponding predicate symbol `'red'`. In addition, we promote the use of an *object classification* procedure in order to semantically categorize and label each perceived object. The convolutional neural network (CNN) architecture that we use for this purpose is based on the *GoogLeNet* model (Szegedy et al., 2015), which we have trained and fine-tuned based on 101 object categories that can be expected to be found in a kitchen domain.

The extracted perceptual and symbolic information for each perceived object is then encapsulated in an internal data structure α_t^x, called an *anchor*, indexed by time t and identified by a unique identifier x (e.g. `'mug-2'`, `'apple-4'`, etc.). The goal of an *anchoring system* is to manage these anchors based on the result of a *matching function* that compares the attribute values of an unknown candidate object against the attribute values of all previously maintained anchors. Anchors are then either created or maintained through two general functionalities:

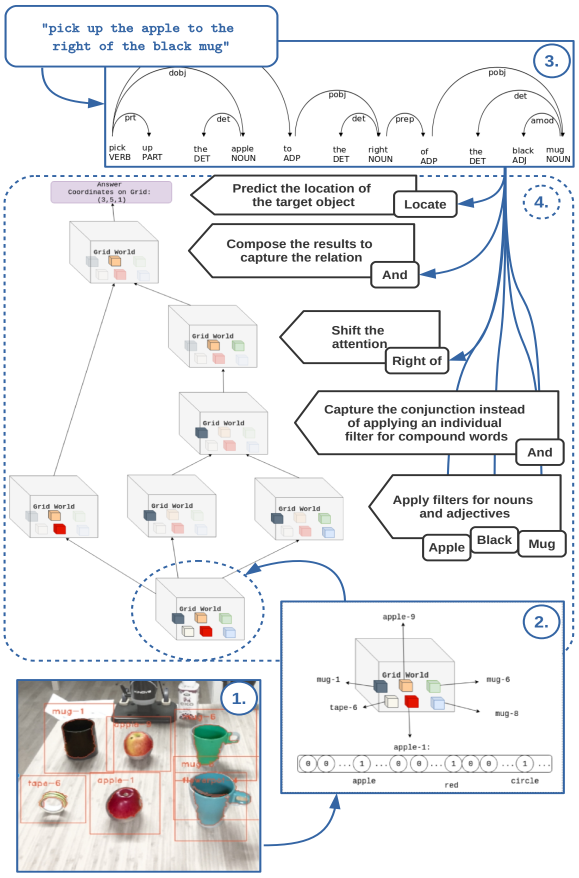
- *Acquire* – initiates a new anchor whenever a candidate object is received that does not match any existing anchor α^x.
- *Re-acquire* – extends the definition of a matching anchor α^x from time $t - k$ to time t. This functionality assures that the percepts pointed to by the anchor are the most recent perceptual (and consequently also symbolic) representation of the object.

However, comparing attribute values of anchored objects and percepts by some distance measure and deciding, based on the measure, whether an unknown object has previously been perceived or not is a non-trivial task. Nevertheless, since anchors are created or maintained through either one of the two principal functionalities *acquire* and *re-acquire*, it is evident that the desired outcome for the combined compared values is a *binary* output, i.e. should a percept be acquired or re-acquired. In previous work on anchoring (Persson et al., 2019), we have therefore suggested that the problem of invoking a correct anchoring functionality is a problem that can be approximated through learning from examples and the use of *classification algorithms*. For this work, we follow the same approach.

2.2 Natural Language Grounding

In this study, we focus on understanding spatial language that includes *pick up* and *place* related verbs, and *referring expressions*. An instruction refers to a target object using its representative features (e.g. color, shape, size). If a noun phrase does not resolve the ambiguity in the world, the instruction resolves the ambiguity by specifying the target object with its relative position to other surrounding objects. This hierarchy tries to bring the attention to finding the unique object, then shifts the attention to the targeted object. Based on this idea, we model the language grounding process as controlling the attention on the world representation by adapting the neural module networks approach proposed by Andreas et al. (2016b).

Our natural language grounder has three components: a preprocessor, an instruction parser and a program executor. Given specific anchor information (Figure 1 – № 1), the preprocessor transforms



"pick up the apple to the right of the black mug"
3.
dobj
prt
det
pobj
det
prep
pobj
det
amod
pick VERB
up PART
the DET
apple NOUN
to ADP
the DET
right NOUN
of ADP
the DET
black ADJ
mug NOUN
4.
Answer Coordinates on Grid: (3,5,1)
Predict the location of the target object
Locate
Compose the results to capture the relation
And
Shift the attention
Right of
Capture the conjunction instead of applying an individual filter for compound words
And
Apply filters for nouns and adjectives
Apple
Black
Mug
Grid World
2.
apple-9
mug-1
mug-6
tape-6
mug-8
apple-1:
apple
red
circle
1.
mug-1
apple
tape-6
apple-1

Figure 1: Demonstration of the language grounding process for the instruction "pick up the apple to the right of the black mug". Anchoring system sends the snapshot of the anchors (1). Then, a preprocessor transforms the anchors into a grid representation which the language grounding system operates on (2). The parser parses the given instruction and generates a computation graph which specifies the execution order of neural modules (3). Finally, the neural modules are executed according to the computation graph to produce the action (4).

the anchor information into an intermediate representation in grid form (Figure 1 – № 2). The instruction parser produces a computational program by exploiting the syntactic representation (Figure 1 – № 3) of the instruction with a dependency parser[1]. The program executor runs (Figure 1 – № 4) the program on the intermediate representation to produce commands.

Preprocessor. The anchoring framework maintains the object descriptions predicted from the raw visual input. To be able to ground the language onto those descriptions, we map the available information (object class, color, size and shape attributes)

[1]https://spacy.io/

to a 4D grid representation. We represent each anchor as a multi-hot vector and assign this vector to a cell where the real world coordinates of the object fall into.

Program Executor. The program generated by the parser is a collection of neural components that are linked to each other depending on the computation graph. The design of neural components reflects our intuition about the attention control. A ***Detect*** module is a convolutional neural network with a learnable filter that captures a noun or an adjective. This module creates an attention map over the input grid.

$$Detect(w, b, x) = relu(w \otimes x + b) \qquad (1)$$

The *Detect* module operates on the original grid input x tensor, where the dimensions are (W, H, L, C). The first three dimensions represent the spatial dimensions and C denotes the length of the feature vector. w is the filter of size $(1, 1, 1, C)$ and b is the bias. $\otimes$ is a convolution operation.

Although a *Detect* module can capture the meaning of a noun phrase (e.g., red book), the model cannot generalize to unseen compound words. To overcome this, we design the *And* module to compose the output of incoming modules. This module multiplies the inputs element-wise in order to calculate the composition of words (e.g., the big red book). Since the incoming inputs are attention maps over the grid world, an ***And*** module produces a new attention map by taking the conjunction of its inputs. In the following equation, the $\odot$ denotes the element-wise multiplication.

$$And(a_1, a_2) = a_1 \odot a_2 \qquad (2)$$

An output of a subgraph for a noun phrase is an attention map that highlights the positions for the corresponding objects that occur. A ***Shift*** module shifts this attention in the direction of the preposition that the module represents. This module is also a convolutional neural network similar to a ***Detect*** module. However, the module remaps the attention instead of capturing the patterns in the grid world.

$$Shift(w, a) = relu(w \otimes a) \qquad (3)$$

The *Shift* module operates on an incoming attention map, where the dimensions are $(W, H, L, 1)$. w is the filter of size of $(2*W+1, 2*H+1, 2*L+1, 1)$. We use the padding to be able to perform the shifting operation over the whole grid. The pad size is the same as the input size.

A ***Locate*** module takes an attention map and produces a probability distribution over cells by applying a softmax classifier for being the targeted object. We use the cell with the highest probability as the prediction. A ***Position*** module gets a source anchor, a preposition and a target anchor, and produces a real world coordinate. It merely calculates the position available in the direction of the preposition from the target anchor, where the source anchor can fit.

Parser. We find the verbs in the instruction along with the subtrees attached to them. For each verb and its subtree, we search for the direct object of the verb. Then we build a subgraph for the direct object and its modifiers. Depending on the verb type, we build different subgraphs. If the verb is ”pick up” related, then we look for the preposition that relates the given noun to another noun. If one is found, then a subgraph is created for the preposition object using the noun phrase that the object belongs to. Finally, the end point of the subgraph is combined with a *Shift* module. For each preposition object, we repeat the same process to handle prepositional phrase chains.

If the verb is ”put” related, we find the preposition that is linked to the verb and the object of the preposition. We build a subgraph that refers to the object of the preposition similar to the ”pick up” case. Finally, there is a *Position* module to produce the coordinates to put the direct object, where the position is referred with the auxiliary objects.

3 System Description

In the upper part of Figure 2, we illustrate our physical *kitchen table* system setup, which consists of the following devices: *1)* a *Kinova Jaco light-weight manipulator* (Campeau-Lecours et al., 2019), *2)* a *Microsoft Kinect2 RGB-D* sensors, and *3)* a dedicated PC with an Intel© Core™ i7-6700 processor and an Nvidia GeForce GTX 970 graphics card.

In addition, we have a modularized software architecture that utilizes the libraries and communication protocols available in the Robot Operating System (ROS)[2]. Hence, each of the modules, illustrated in the lower part of Figure 2, consists of one or several individual subsystems (or ROS nodes). For example, the *visual object anchoring* module consists of the following subsystems: 1) a *perceptual system*, 2) a *symbolic system*, and 3) an *anchoring system*. For a seamless integration be-

[2]http://www.ros.org/

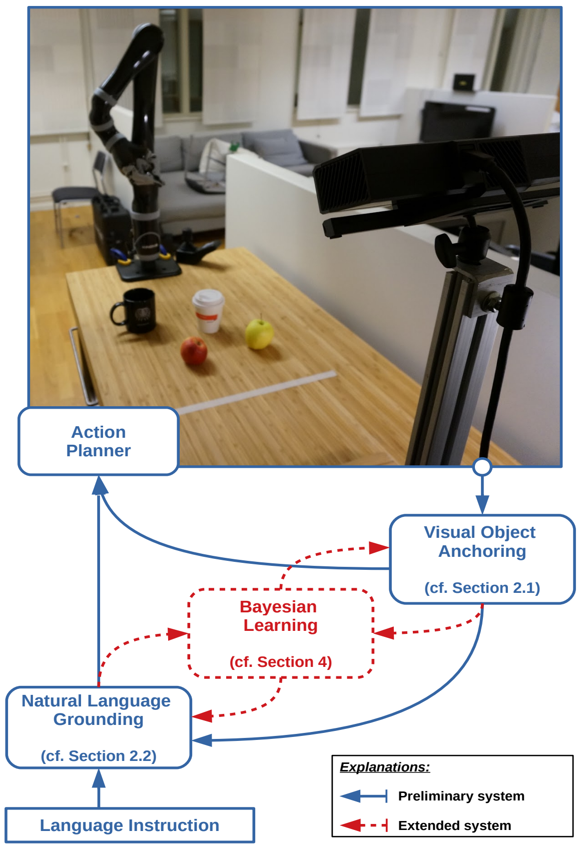


Figure 2: A depiction of both used physical system setup (upper), as well as used software architecture (lower). The arrows represent the flow of data between the modules of the software architecture. Blue solid arrows and boxes illustrate the preliminary system (outlined in Section 2), while red dashed arrows and boxes illustrate the novel extension for reasoning about different symbolic label configurations (and hence resolving inconsistencies between language and perception), by using Bayesian learning (as presented in Section 4).

tween software and hardware, we are further taking advantage of both the *MoveIt! Motion Planning Framework*[3], as well as the *ROS-Kinect2 bridge* developed by (Wiedemeyer, 2014 – 2015). The *MoveIt! "planning scene"* of the *action planner* for the robot manipulator, as well as the grid world representation used by the language grounding system (cf. Section 2.2), are, subsequently, both populated by the same updating anchoring representations (cf. Section 2.1). Hence, the visual sensory input stream is indirectly mapped to both objects considered in the dialogue by the language grounder, as well as the objects upon which actions are executed.

[3]https://moveit.ros.org/

4 Resolving Inconsistencies

Based purely on the perceptual input, the anchoring system produces a probability distribution $p(l)$ over the possible labels (e.g. $[0.65 : \text{apple}, 0.35 : \text{pear}]$) for each anchor. We are now interested in the probability of a label l for an anchor given a natural language instruction i and the grounding g of that instruction in the real world. This is the conditional probability $p(l \mid g, i)$. We introduce, furthermore, the notion of a label configuration c. This is easiest explained by an example: imagine having two anchors and each of the anchors has two possible labels, then there are 2×2 possible label configurations. A label configuration is, hence, a label assignment to all the anchors present in the scene.

Now we need to transform the conditional probability into a function that is computable by the anchoring system and the language grounder. The first steps (Equations 4-6) are quite straight forward and follow basic probability calculus.

$$\begin{aligned} p(l|g,i) &= \textstyle\sum_c p(l,c|g,i) & (4)\\ &= \textstyle\sum_c p(l|c,g,i)p(c|g,i) & (5)\\ &= \textstyle\sum_c p(l|c)p(c|g,i) & (6)\end{aligned}$$

In Equation 6 we assume that g and i are conditionally independent of the label of an anchor given the label configuration c. This can be seen in the following way. Imagine two anchors with two possible labels each. Given that we are in a specific label configuration, we immediately know what label the single anchors have. This means that the probability of a label for an anchor is 1 if it matches the label in the configuration and 0 otherwise. This reasoning is independent of the grounding and the instruction.

We have now split up the labels (produced by the anchoring system) and the grounding into two factors, which can be calculated separately. The first one can be calculated as follows:

$$p(l \mid c) = \frac{p(l,c)}{p(c)} = \frac{\prod_{j \in c} p(l_j)}{N_c} \qquad (7)$$

This is the product of the probabilities of the labels that constitute a label configuration divided by the number of configurations. Assuming a uniform distribution over the label configurations (division by N_c) is equivalent to assuming that each possible label configuration is equally likely *a priori*. This means that we make no assumption about which class of objects occur more regularly or which class of objects (of the 101 possible classes) occur more often together with other classes of objects.

We tackle now the second factor in Equation 6. Equation 8-11 are again straightforward probability calculus. In Equation 12 we assume that the label configuration and the instruction are independent: their probabilities factorize. In Equation 13 the probabilities of i cancel out and we assume again a uniform distribution for the label configurations (cf Equation 7). In Equation 14 we then have a numerator and denominator that are expressed in terms of $p(g \mid c, i)$, which is exactly the function approximated by our neural language grounding system, cf. subsection 2.2.

$$\begin{aligned}
p(c|g,i) &= \frac{p(c,g,i)}{p(g,i)} && (8)\\
&= \frac{p(g|c,i)p(c,i)}{p(g,i)} && (9)\\
&= \frac{p(g|c,i)p(c,i)}{\sum_c p(c,g,i)} && (10)\\
&= \frac{p(g|c,i)p(c,i)}{\sum_c p(g|c,i)p(c,i)} && (11)\\
&= \frac{p(g|c,i)p(c)p(i)}{\sum_c p(g|c,i)p(c)p(i)} && (12)\\
&= \frac{p(g|c,i)^1/N_c}{\sum_c p(g|c,i)^1/N_c} && (13)\\
&= \frac{p(g|c,i)}{\sum_c p(g|c,i)} && (14)
\end{aligned}$$

Plugging Equations 7 and 14 back into Equation 6 gives the learned probability of the label l of an anchor given the instruction i and the grounding g of that instruction.

$$p(l \mid g, i) = \frac{\sum_c (\prod_{j \in c} p(l_j)) p(g|c,i)}{N_c \sum_c p(g|c,i)} \qquad (15)$$

As mentioned in Section 2.1 the anchoring system encapsulates 101 object categories, which means that the anchoring system produces a categorical probability distribution over 101 different labels for each anchor. With only two anchors this results in already 101^2 different configurations. It is easy to see that computing $p(l \mid g, i)$ (cf. Equation 15) suffers from this curse of dimensionality. Therefore, we limited ourselves to the two labels with the highest probability per anchor, in the experiments too. This gives 2^{N_A} possible configurations, with N_A being the number of anchors present.

5 Experiments

5.1 Synthetic Data

Data demanding nature of neural networks requires large amounts of data to generalize well. Artificial data generation is one way of generating such datasets (Andreas et al., 2016b; Kuhnle and Copestake, 2017; Johnson et al., 2016). Therefore, we designed a series of artificial learning tasks before applying the model to a real-world problem. In each task, we generate a random grid world that provides the necessary complexity and ambiguity that fit the scenario. First, an object is placed on the grid world and decorated with attributes randomly as the target object. Then depending on the scenario, an auxiliary object and distractors (objects that have similar attributes as the target object) are placed on the grid world. We also generate objects that are not related to the target (or auxiliary object) to introduce additional noise. We limit the total number of objects to 10. We set the number of distractors as 2 in the experiments. Finally, we generate the ground truth computation graph for composing neural modules. We list the scenarios below in increasing order of difficulty (i.e., a combination of the ambiguity present in the grid world and the number of language components involved).

1. Using the **name** of a targeted object in the instruction is enough to localize the targeted object.

2. There is more than one object that has the same category with a targeted object. To solve the ambiguity, one or more discriminative **adjective(s)** are used.

3. The same world configuration as the second one. To solve the ambiguity, the object is described with a **prepositional phrase** that utilizes a single referent object.

4. The same world configuration as the third. **Adjectives** are used to describe a targeted object in addition to a **prepositional phrase**. In this case, adjectives are unnecessary, but the scenario measures whether additional components bring noise or not.

5. All other objects that have the same category with a targeted object have the same set of features as the targeted object has. Hence, the targeted object is only distinguishable by its position. To solve the ambiguity, the object

is described with a **prepositional phrase** that utilizes a referent object along with necessary **adjectives**.

6. It is a **random** scenario from the above list.

5.2 Training

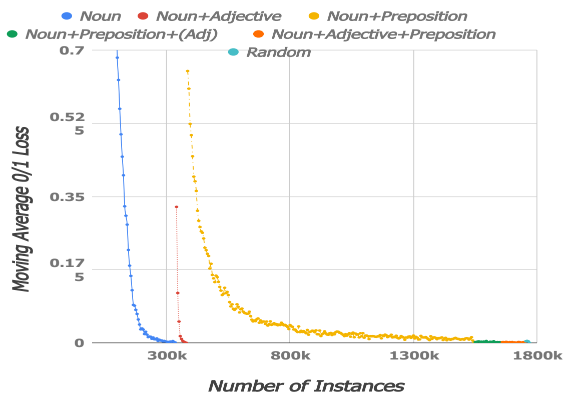


Figure 3: Learning curve of the neural modules.

To be able to measure the compositionality of learned modules, we have two different settings for the data generation. For training, we constrain the 75% of possible attributes for an object class and locations on the grid world that an instance of that object class can present. During testing, we use unconstrained samples generated for the same scenario. This way, we can evaluate if the model infers unseen word compositions, e.g. inferring red mug after seeing red book and black mug in the training time. We follow a curriculum schema to train our modules. Starting from the first scenario described in Section 5.1, we train the model on a stream of constrained randomly generated samples. We evaluate the model periodically on unconstrained samples generated for each period and continue training until the moving average error on the test data falls under a threshold (e.g. 1e-5 in our experiments). We then continue to train the model for the next scenario using learned weights. We set the number of nouns, adjectives and prepositions as 102, 26, and 27, respectively to match with the anchoring system. We use Adam (Kingma and Ba, 2014) with default parameters (i.e. $lr = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$) for the optimization.

Figure 3 presents the learning curve of the model. The third graph (yellow) demonstrates that learning prepositions requires more data as compared to learning nouns (first graph) or adjectives (second graph). The reason for this behavior is twofold. First, the *Shift* modules have more weights to be learned than the *Detect* modules. Second, while the *Detect* modules have a one to one mapping between input and output, the *Shift* modules have many to many relations. There might be more than one active area in the input of a *Shift* module. Since it needs to remap highlighted areas on the grid to other areas , it needs to see different examples that occur in different parts of the grid world in order to learn to ignore the position of the active area.

The remaining graphs show the effectiveness of our design to compose learned modules. Since we do not train any modules from scratch, we can handle the composition of nouns, adjectives and prepositions effectively. Since it is the first time we train all components together in scenario 4, the training requires more data than one would expect when compared with graphs 5 (orange) and 6 (cyan).

6 Showcase

We now proceed with a demonstration of the integrated system: we have a Kinect camera that observes the world, the anchoring representation that builds up a representation of the world based on the raw image data, the language grounder that takes as input a natural language instruction and a probabilistic reasoning component that resolves possible inconsistencies between the robot's world representation and the instruction.

The physical setup up is identical to the one depicted in the image in Figure 2: the robot arm is mounted on the opposite site of a kitchen table of the Kinect camera. The natural language instruction is passed to the language grounder via an *instruction prompt*. In each of the four panels in Figure 4, the instruction prompt is seen at the bottom as rectangular box. We further describe the scenario in the caption of Figure 4.

7 Related Work

Our work is related to two research domains: modular neural nets for language grounding and human-robot interaction for handling ambiguities in one or more modalities. Andreas et al. (2016b,a) introduced neural module networks for visual question answering. Johnson et al. (2017); Hu et al. (2017a) developed policy gradient based approaches to learn to generate layouts instead of using a dependency parser based method. Hu et al. (2017b); Yu et al. (2018); Cirik et al. (2018) applied modular neural networks approach on 'Referring Expres-

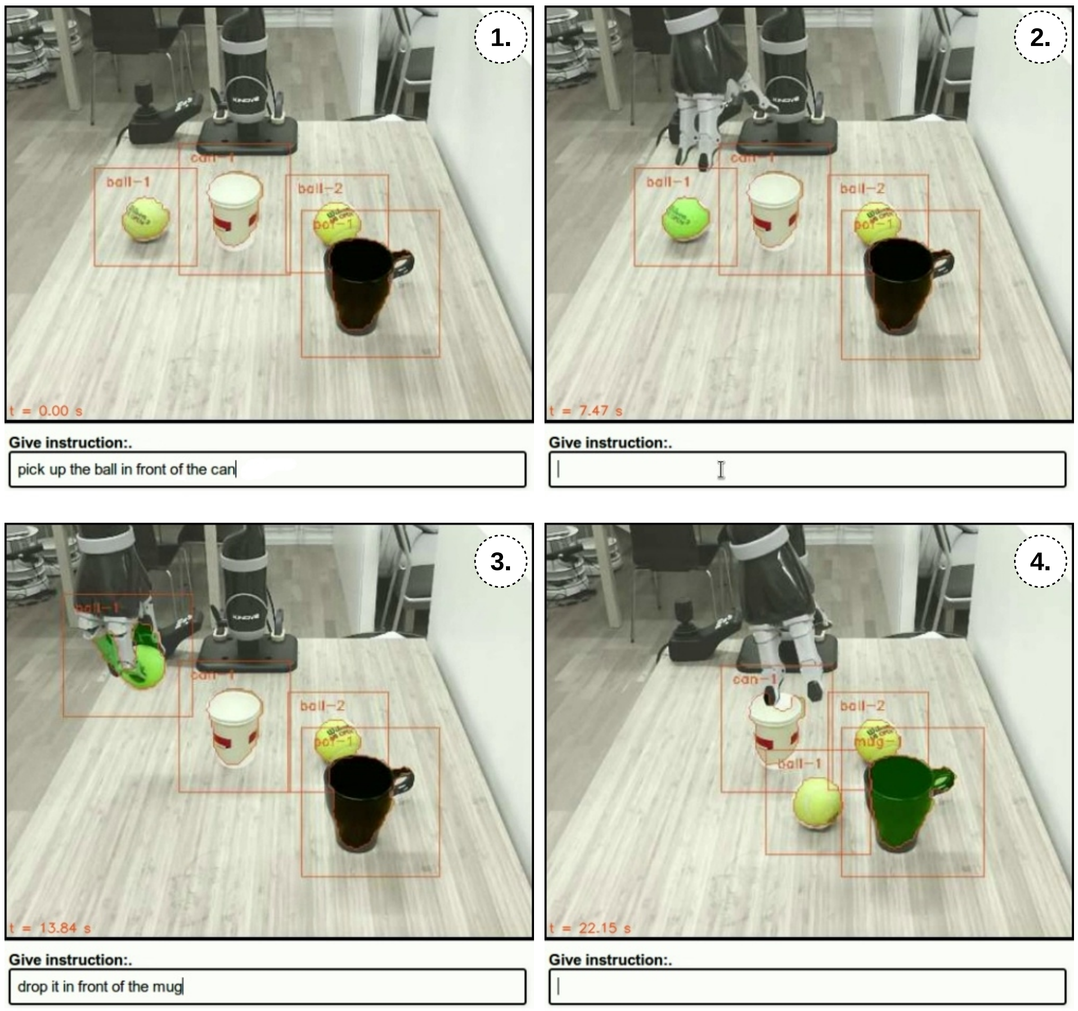


Figure 4: We give the robot the instruction: *"pick up the ball in front of the can"*. The robot executes the action and waits for further instructions. We then give the instruction to *"drop it in front of the mug"*. The problem in this step is that there is no object classified as `'mug'`, which means that none of the objects has as label with highest probability `'mug'`. We correct for this through probabilistic reasoning over not only the top label for each object but a number of top ranked labels per object. This allows the anchoring system to correct its classification of an object based on what we as humans think an object is. Given the instruction, the anchoring system re-classifies the black object from `'pot'` to `'mug'`. The instruction is then successfully carried out. The recorded video can be found here: `https://vimeo.com/302072685`.

sion Understanding‘ task. Das et al. (2018) demonstrated the usage of neural module networks in decision taking in a simulated environment. To our knowledge, the present study is the first work that uses neural module networks approach in the real-world robotic setting.

Learning from human interaction has been extensively studied. Lemaignan et al. (2011, 2012) developed a cognitive architecture that makes decisions by using symbolic information provided as facts (pre-defined) or extended via human-robot dialogues. When compared with our system, their system neither operates on the sensory input nor deals with the uncertainty in the world. Tellex et al. (2013) proposed a system to ask questions to disambiguate the ambiguities presented in the instructions. The robot decides the most ambiguous part of the command which is defined based on a metric derived from entropy and asks questions about it to reduce the uncertainty. They update the generalized grounding graph (Kollar et al., 2013) with answers obtained from the user and use these to perform inference. In contrast, we fix the ambiguity present in the perceptual data. She and Chai (2017) proposed a system to learn to ask questions during the learning of verb semantics. They work on the *Tell me Dave environment* (Misra et al., 2014). The work represents the environment as grounded state fluents (i.e. a weighted logic representation). In this work, language grounding is modeled as the difference between before and after state for an action sequence. They modeled the interactive learning as an MDP and solved it with reinforcement learning.

Thomason et al. (2015) proposed a system that learns the meaning of natural language commands through human-robot dialog. They represent the meaning of instructions with λ-calculus semantic representation. Their semantic parser starts with an initial knowledge and learns through training examples generated by the human-robot conversations. Their dialog manager is a static policy which generates questions from a discrete set of action, patient, recipient tuples. Padmakumar et al. (2017) improved this work with a learnable dialog manager. They train both the dialog manager and the semantic parser with reinforcement learning. This approach was further extended in (Thomason et al., 2019), where the authors combine the approach in Thomason et al. (2015) and Thomason et al. (2017) to obtain a system that is capable of concept acquisition through clarification dialogues. Instead of asking questions, we implicitly fix the perception with the information hidden in instructions. A further difference to these works is that we learn the language component in a simulated offline step, whereas they deploy active online learning, starting from a limited initial vocabulary.

This is also related to the work of Perera and Allen, who present a system that tries to emulate child language learning strategies by describing scenes to a robot agent, which has to learn actively new concepts. The authors deploy probabilistic reasoning to manage erroneous sensor readings in the vision system. Apart from the active learning approach, there is also a conceptual difference: in our work, we do not consider discrepencies between the perceptual system (anchoring) and the language grounder as errors in the perceptual system but simply as different models of the world.[4]

As mentioned in Section 1, the work related closest to our approach is presented in Mast et al. (2016). The authors base their work on geometric *conceptual spaces* (Gärdenfors, 2004), which situates their work in the sub-domain of *top-down anchoring* (Coradeschi and Saffiotti, 2000). The geometric conceptual spaces induce a probabilistic model-based language grounder. This enables a robot to reason probabilistically over a description of a scene, given by an other agent, and single out the object that is most likely being referred to. In contrast, we present an approach to perform Bayesian learning over a learned language grounding model and a bottom-up anchoring approach.

[4]This view taps into the philosophical question of whether one can ever truly know the nature of an object, cf. *thing-in-itself* (Kant, 1878), for which we omit a discussion.

8 Conclusions and Future Work

We introduced the problem of belief revision in robotics based solely on implicit information available in natural language in the setting of sensor-driven bottom-up anchoring in combination with a learned language grounding model. This is in contrast to prior works, which study either explicit information or are based on top-down anchoring. We proposed a Bayesian learning approach to solve the problem and demonstrated its validity on a real world showcase involving computer vision, natural language grounding and robotic manipulation.

In future work we would like to perform a more quantitative analysis of our approach to which end it is imperative to circumvent the curse of dimensionality emerging in the Bayesian learning step (cf. Equation 15). It would also be interesting to investigate whether our approach is amenable to natural language other than instructions.

A main limitation of our current approach is the limited size of the predefined vocabulary. It would be more practicable if a robot were able to extend its vocabulary through the interaction with a human, i.e. through dialogue. A possible solution would be to learn a probabilistic model (which resolves inconsistencies between language and vision) that takes into account the possible of currently unknown vocabulary occurring. Such an approach would still allow us to learn the anchoring of objects and the language grounding separately, while learning a much richer model to resolve inconsistencies than the one described in this work.

Acknowledgements

This work has been supported by the ReGROUND project (http://reground.cs.kuleuven.be), which is a CHIST-ERA project funded by the EU H2020 framework program, the Research Foundation - Flanders, the Swedish Research Council (Vetenskapsrådet), and the Scientific and Technological Research Council of Turkey (TUBITAK). The work is also supported by Vetenskapsrådet under the grant number: 2016-05321 and by TUBITAK under the grants 114E628 and 215E201.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016a. Learning to compose neural networks for question answering. *arXiv preprint arXiv:1601.01705*.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016b. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 39–48.

Alexandre Campeau-Lecours, Hugo Lamontagne, Simon Latour, Philippe Fauteux, Véronique Maheu, François Boucher, Charles Deguire, and Louis-Joseph Caron L'Ecuyer. 2019. Kinova modular robot arms for service robotics applications. In *Rapid Automation: Concepts, Methodologies, Tools, and Applications*, pages 693–719. IGI Global.

Volkan Cirik, Taylor Berg-Kirkpatrick, and Louis-Philippe Morency. 2018. Using syntax to ground referring expressions in natural images. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

S. Coradeschi and A. Saffiotti. 2000. Anchoring symbols to sensor data: preliminary report. In *Proc. of the 17th AAAI Conf.*, pages 129–135, Menlo Park, CA. AAAI Press. Online at http://www.aass.oru.se/~asaffio/.

Abhishek Das, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. 2018. Neural modular control for embodied question answering. *arXiv preprint arXiv:1810.11181*.

J. Elfring, S. van den Dries, M.J.G. van de Molengraft, and M. Steinbuch. 2013. Semantic world modeling using probabilistic multiple hypothesis anchoring. *Robotics and Autonomous Systems*, 61(2):95–105.

Peter Gärdenfors. 2004. *Conceptual spaces: The geometry of thought*. MIT press.

Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. 2017a. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 804–813.

Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. 2017b. Modeling relationships in referential expressions with compositional modular networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1115–1124.

Drew A Hudson and Christopher D Manning. 2018. Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067*.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2016. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *arXiv preprint arXiv:1612.06890*.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2989–2998.

Immanuel Kant. 1878. *Prolegomena zu einer jeden Künftigen Metaphysik: die als Wissenschaft wird auftreten konnen*. Verlag von Leopold Voss.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas Kollar, Stefanie Tellex, Matthew R Walter, Albert Huang, Abraham Bachrach, Sachi Hemachandra, Emma Brunskill, Ashis Banerjee, Deb Roy, Seth Teller, et al. 2013. Generalized grounding graphs: A probabilistic framework for understanding grounded language. *JAIR*.

Alexander Kuhnle and Ann Copestake. 2017. Shapeworld-a new test methodology for multimodal language understanding. *arXiv preprint arXiv:1704.04517*.

Séverin Lemaignan, Raquel Ros, Rachid Alami, and Michael Beetz. 2011. What are you talking about? grounding dialogue in a perspective-aware robotic architecture. In *2011 RO-MAN*, pages 107–112. IEEE.

Séverin Lemaignan, Raquel Ros, E Akin Sisbot, Rachid Alami, and Michael Beetz. 2012. Grounding the interaction: Anchoring situated discourse in everyday human-robot interaction. *International Journal of Social Robotics*, 4(2):181–199.

A. Loutfi, S. Coradeschi, and A. Saffiotti. 2005. Maintaining coherent perceptual information using anchoring. In *Proc. of the 19th IJCAI Conf.*, pages 1477–1482, Edinburgh, UK.

Vivien Mast, Zoe Falomir, and Diedrich Wolter. 2016. Probabilistic reference and grounding with pragr for dialogues with robots. *Journal of Experimental & Theoretical Artificial Intelligence*, 28(5):889–911.

Dipendra K Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. 2014. Tell me dave: Contextsensitive grounding of natural language to mobile manipulation instructions. In *in RSS*. Citeseer.

Aishwarya Padmakumar, Jesse Thomason, and Raymond J Mooney. 2017. Integrated learning of dialog strategies and semantic parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Ian Perera and James Allen. 2015. Quantity, contrast, and convention in cross-situated language comprehension. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 226–236.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Andreas Persson, Pedro Zuidberg Dos Martires, Amy Loutfi, and Luc De Raedt. 2019. Semantic relational object tracking. *arXiv preprint arXiv:1902.09937*.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. 2017. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976.

Lanbo She and Joyce Chai. 2017. Interactive learning of grounded verb semantics towards human-robot communication. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1634–1644.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9.

Stefanie Tellex, Pratiksha Thakerll, Robin Deitsl, Dimitar Simeonovl, Thomas Kollar, and Nicholas Royl. 2013. Toward information theoretic human-robot dialog. *Robotics*, page 409.

Jesse Thomason, Aishwarya Padmakumar, Jivko Sinapov, Justin Hart, Peter Stone, and Raymond J Mooney. 2017. Opportunistic active learning for grounding natural language descriptions. In *Conference on Robot Learning*, pages 67–76.

Jesse Thomason, Aishwarya Padmakumar, Jivko Sinapov, Nick Walker, Yuqian Jiang, Harel Yedidsion, Justin Hart, Peter Stone, and Raymond J Mooney. 2019. Improving grounded natural language understanding through human-robot dialog. *arXiv preprint arXiv:1903.00122*.

Jesse Thomason, Shiqi Zhang, Raymond J Mooney, and Peter Stone. 2015. Learning to interpret natural language commands through human-robot dialog. In *IJCAI*, pages 1923–1929.

Thiemo Wiedemeyer. 2014 – 2015. IAI Kinect2. `https://github.com/code-iai/iai_kinect2`. Accessed February 28, 2019.

Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. 2018. Mattnet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1307–1315.