Learning Instance-Level N-Ary Semantic Knowledge At Scale For Robots Operating in Everyday Environments

Author Names Omitted for Anonymous Review. Paper-ID 120

Abstract—**Modeling n-ary relations between semantic properties of objects is crucial for accurately inferring unknown properties of objects in different contexts. Yet most existing approaches model solely binary relations. We introduce an approach that uses a contextualized language model to efficiently learn n-ary relations between object properties. We also contribute LINK, a unique dataset containing n-ary object properties of 1457 object contexts with 16 properties types and 202 total properties. We quantitatively validate our approach against five prior methods on LINK. Compared to prior state of the art, our model obtains 10% improvement in predicting unknown properties of new object instances while reducing training and inference time by 166 times. Additionally, we apply our work to a mobile manipulation robot, demonstrating its ability to leverage n-ary reasoning to retrieve objects and actively detect object properties.**

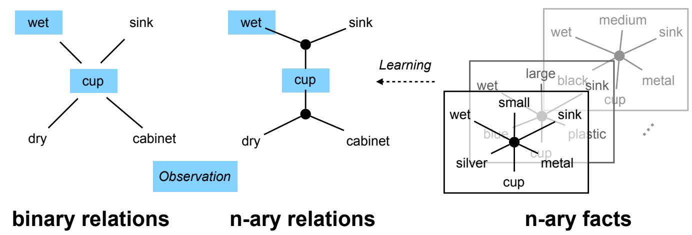

Fig. 1: N-ary representation of object knowledge allows robots to reason about interrelations between object properties and therefore provide more accurate information. In our framework, we learn generalizable n-ary relations from object instances represented as n-ary observations.

I. Introduction

A key challenge for robust operation in everyday human environments is the need to effectively model a wide range of objects – to predict object locations, properties, and uses. Semantic reasoning techniques enable robots to encode and use semantic knowledge about objects (e.g., *cups* are usually *ceramic*, located in *kitchens*, and used for *drinking*), which aid robots in performing many real-world tasks, such as inferring missing information in human instructions [42, 10], efficiently searching for objects in homes [56, 55], and manipulating objects based on their affordances and states [2, 32].

Prior work on semantic object representations has encoded semantic object properties primarily as pairwise relations between an object's *class* label and its semantic *properties* (e.g., the *cup* is *wet*, the *cup* is *in cabinet*) [13, 11, 58, 48, 44] (Figure 1 left). However, by ignoring the inherent relationship between object properties, such reasoning systems can be brittle, failing to take advantage of key information. For example, observing the *cup* is *wet* fails to infer that the *cup* is more likely to be located *in sink* than *in cabinet*.

In this work, we address the problem of predicting semantic properties of objects based on partial observations. We introduce a novel semantic reasoning framework that uses n-ary relations to model complex, inter-related object properties while enabling the ability to reason at different levels of abstraction (Figure 1 middle). For example, a robot searching for a *cup*, with no additional attributes, is able to perform class-level inference to identify both the *cabinet* and *sink* as likely locations. However, given additional properties, such as *wet*, n-ary relations enable more refined reasoning and the ability to detect that wet cups are more commonly found *in sink* rather than *in cabinet*.

A key challenge presented by n-ary representations is the collection of semantically meaningful n-ary data, which requires object properties to be conditioned on other properties. In fact, all semantic reasoning frameworks used in robotics to date have utilized binary representations [13, 11, 58, 48, 44]. In this work, we obtain n-ary observations each representing an object instance within a particular environmental context (e.g., a *small silver metal cup* that is *wet* and *in sink*), from which we then learn a model capturing generalizable n-ary relations (Figure 1 right). Since the n-ary relations are learned from object instances, they also encode knowledge at the instance-level. To mine generalizable patterns from n-ary observations, we introduce a neural network model inspired by recent advances in contextualized language models [15, 51]. Our model is trained to predict hidden properties of new object instances. With this training objective, our model naturally learns to generalize n-ary relations.

In summary, our work contributes:

- an n-ary instance-level representation of objects, which enables modeling n-ary relations between object properties and variance between object instances,
- a neural network model which leverages the Transformer [51] to learn semantic knowledge about objects from data,
- LINK dataset consisting of 1457 object instances associated with 16 property types and 202 properties, the richest situated object dataset to date.

We quantitatively validate our approach against five prior methods on the above dataset and demonstrate that our representation and reasoning methods lead to significant improvement in predicting unknown properties of new object instances over prior state of the art while reducing computation time. Additionally, we apply our work to a mobile manipulation robot, demonstrating its ability to leverage n-ary reasoning to retrieve objects and actively detect object properties.

II. Related Work

Our work is related to the following prior efforts.

A. Semantic Reasoning in Robotics

Many ontologies and knowledge graphs have been used across AI and robotics to encode general knowledge about objects (e.g., locations, properties, uses, and class hierarchies) [31, 44, 48, 30, 50]. In robotics, a key challenge for semantic reasoning is generalization to previously unseen scenes or environments. Bayesian logic networks have been used to cope with noise and non-deterministic data from different data sources [11]. More recently, knowledge graph (KG) embedding models were introduced as scalable frameworks to model object knowledge encoded in multi-relational KGs [13, 3]. Although the above techniques effectively model objects, they only support reasoning about binary class-level facts, therefore lacking the discriminative features needed to model object semantics in realistic environments.

Other frameworks take a learning approach to modeling object semantics. Methods for learning relations between objects, between object properties, and between objects and their environments have shown to be beneficial for detecting objects on table tops [28, 22, 41], finding hidden objects in shelves [38], predicting object affordances [58], and semantic grasping [2, 32]. However, most methods [41, 38, 58, 2] leverage probabilistic logic models to learn these relations, which have scalability issues that limit them from modeling inter-connected relations in larger domains. In contrast, our proposed framework learns n-ary relations between 16 property types and over 200 properties, the richest representation to date.

Our approach is also related to methods for modeling objects from metric data. In computer vision, object attributes are extracted from images [18, 17, 46]. Recent techniques in visual question answering [36, 39] and language grounding [45, 25] allow robots to answer questions about objects and describe objects in natural language sentences. Multimodal perception has also helped robots interpret salient features of objects in different modalities [16, 34, 19]. Interactive perception can further leverage a robot's exploratory actions to reveal sensory signals that are otherwise not observable [12, 49, 7]. We consider our approach complimentary to the above, as our framework can leverage the rich semantic information extracted from these methods to infer additional unknown object properties.

B. Modeling N-Ary Facts

Our neural network model is closely related to methods developed in the knowledge graph community. Many relational machine learning techniques, including most recent transformer models [52, 8], have been developed for modeling KGs and in particular predicting missing links in KGs [40]. These techniques treat a KG as set of triples/binary facts, where each triple (h, r, t) links two entities h and t with a relation r (e.g., *(Marie Curie, educated at, University of Paris)*). Despite the wide use of triple representation, many facts in KGs are hyper-relational. Each hyper-relational fact has a base triple (h, r, t) and additional key-value (relation-entity) pairs (k, v) (e.g., $\{$*(academic major, physical), (academic degree, Master of Science)*$\}$). A line of work converts hyper-relational facts to n-ary meta-relations $r(e_1, ..., e_n)$ and leverages translational distance embedding [53, 57], spatio-translational embedding [1], tensor factorization [33] for modeling. Other approaches directly learn hyper-relational facts in their original form using various techniques, including convolutional neural networks, graph neural networks, and transformer models [43, 20]. A representation of hyper-relational facts more closely related to our work is used by [21]. This approach unifies n-ary representation by converting the base triple to key-value pairs; it uses convolutional neural network for feature extraction and then models relatedness of role-value pairs with a fully connected networks. In our work, we model facts with much higher arities than existing work in the KG community and directly reason about n-ary relations between role-value pairs using the transformer model.

III. Problem Definition

Given a set of observed/known object properties, we aim to predict an unobserved/unknown property of a novel situated object instance using semantic knowledge learned from data. We define a *situated object instance* as a particular specimen of a given object class within a particular environmental context (e.g., the full red Solo cup on the kitchen counter). The object's semantic representation encodes both its immutable properties (e.g., class, material, shape, and fragility) and mutable properties (e.g., location, fullness).

We use the n-ary representation to model all object data. Each n-ary relation is defined by a set of role-value pairs $\{r_i : v_i\}$, where $r_i \in \mathcal{R}$ is the role set, $v_i \in \mathcal{V}$ is the value set, and $i = 1, ..., n$. The value n represents the arity of the relation. In the context of modeling object semantics, each role corresponds to a property type and each value corresponds to a property value. In this representation, our task can be formally written as $\{r_1 : v_1, ..., r_{n-1} : v_{n-1}, r_n :?\}$, where $n-1$ is the number of known properties, and r_n is the type of the property being queried. The number of known properties n determines the level of abstraction for the query. A smaller n queries more abstract semantic knowledge (e.g., $\{$*class: cup, material: ?*$\}$) and a larger n queries more specific semantic knowledge (e.g., $\{$*class: cup, transparency: opaque, physical property: hard, color: brown, material: ?*$\}$).

IV. LINK Dataset

In this section, we present the content and features of the LINK dataset. Our dataset contains 1457 fully annotated situated object instances. In Table I, we compare the content of our dataset to currently existing data sources; as can be

TABLE I: Comparing available dataset for learning object semantics.

Dataset	Application	# Object Classes	# Object Instances	# Properties	# Property Types	Situated	Complete Annotation
Shop-VRB [39]	Vision & Language	20	66	99	6		✓
GoLD [25]	Language Grounding	47	207	/	/		
Thomason 2016 [49]	Interactive Perception	4	32	81	6	✓	✓
Zhu 2014 [58]	Knowledge Base	40	4000	97	4		✓
Paolo 2019 [2]	Semantic Grasping	8	30	44	5		✓
AI2Thor [27]	Simulation	/	125	28	6		✓
Ours	Semantic Reasoning	11	98	202	16	✓	✓

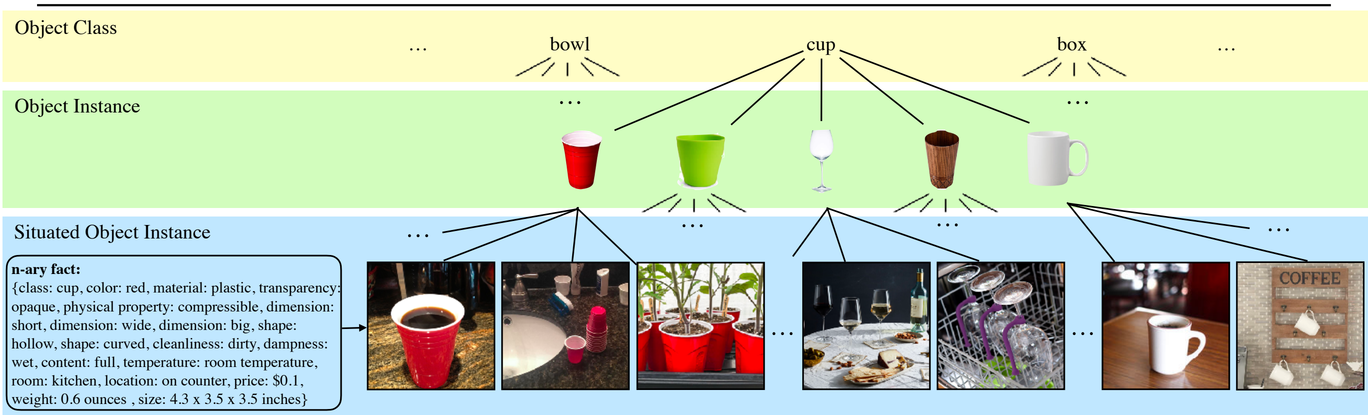

Fig. 2: An example of the collected data showing various instances of cups and diverse environmental contexts each of these instances can be found in. Pictures at the situated object instance level are for illustration but correspond to descriptions of the situations in our dataset. Each situated object instance has a fully annotated n-ary observation (bottom left).

seen, our dataset has the most diverse set of property types and property values, leading to much richer and more realistic object representations. Our dataset preserves variance between object instances by having on average of nine objects per class. Objects in different situations are captured by properties such as location, cleanliness, temperature, dampness, and etc. Furthermore, our dataset provides complete and logically coherent annotations (truth values) of all properties for each situated object instance. Figure 2 illustrates the hierarchy of objects in our dataset, which facilitates the learning of generalizable n-ary relations between object properties at different levels of abstraction.

TABLE II: Object properties and types in our dataset.

Type (# Value)	Values
class (11)	bottle, bowl, box, brush, can, cup, fork, ladle, pan, spatula, sponge
material (8)	ceramic, foam, glass, metal, paper, plastic, porcelain, wood
transparency (3)	opaque, translucent, transparent
dimension (10)	big, deep, long, narrow, shallow, short, small, thick, thin, wide
physical property (6)	absorbent, compressible, elastic, fragile, hard, soft
shape (9)	angular, blunt, curved, flat, forked, hollow, irregular, sharp, straight
*temperature (3)	cold, hot, room temperature
*fullness (3)	empty, full, half
*dampness (3)	damp, dry, wet
*cleanliness (3)	clean, dirty, normal
price (3)	cheap, expensive, medium
weight (3)	heavy, light, medium
size (3)	large, medium, small
*room (11)	balcony, bathroom, bedroom, childś room, closet, dining room, garage, kitchen, laundry, living room, study
color (15)	black, blue, bronze, brown, clear, colorful, gold, green, orange, pink, purple, red, silver, white, yellow
*location (117)	in bag, in basket, in bathtub, in bin, in box, in bucket, in cabinet, in cooler, on bathtub, on bed, on bench, on bookshelf, ...

A. *Objects and Properties*

Our dataset contains 98 instances of everyday household objects organized into 11 object classes. For each object class, we selected objects diverse in sizes, geometries, materials, visual appearances, and affordances from the Amazon product website. We created the initial set of 85 properties (the remaining 117 location properties are crowdsourced) from the collection of 423 adjectives that people use for describing objects [35]. We also followed GermaNet[1] [23] to categorize these properties into 16 distinct types based on their semantic meanings. Table II shows the property values and types in our dataset (mutable properties are labeled with asterisk).

B. *Collection of N-ary Labels*

Given 98 object instances and 16 property types, our next step was to generate 1457 unique situated object instances in which an object is described by a semantically meaningful combination of properties. We used Amazon Mechanical Turk (AMT) to crowdsource property combinations. Specifically, we extracted pictures of each object, as well as details of its material, weight, dimension, and price, from an Amazon product web page. We then conducted a three-stage crowdourcing process. First, for all 98 object instances, we showed a picture of the object to AMT workers and asked them to list the object's immutable properties. Second, we presented AMT workers with an object and a room, and had them write down three situations in which that object-room combination could be encountered, including details of the location of the object, the associated daily activity, and the object state (e.g., a wet cup on the bathroom counter used for rinsing after brushing teeth). Third, we presented a new set of AMT workers with the above collected situated object descriptions, and had them label mutable properties (e.g., wet, empty, clean) for the associated object. To ensure the quality of the

[1]GermaNet, the German version of the English lexical database Wordnet [37], provides hierarchical structures for adjectives.

crowdsourced data, we used 3 annotators for each question and filtered workers based on gold standard questions. We manually verified descriptions of situations from stage 2.

V. Approach

Given $n-1$ properties, we aim to predict the the n^{th} property of type r_n, i.e., $\{r_1 : v_1, ..., r_{n-1} : v_{n-1}, r_n :?\}$, where $n-1$ is the number of observed properties. As shown in Figure 3, our model takes $\{r_1 : v_1, ..., r_{n-1} : v_{n-1}, r_n :$ [MASK]$\}$ as input, where [MASK] is a special token for the query property. The masked input is then fed into the transformer encoder [51], which builds a contextualized representation of the input. Finally, The encoding at the n^{th} position is used to predict the query property via a feedforward and a softmax layer. Now we describe each component of the model and discuss how to train the model to learn n-ary relations between object properties.

A. Input Encoder

The input encoder uses learned embeddings to convert role-value pairs in the input to vectors of dimension d_{model}. Specifically, for each pair, we construct its representation as

$$h_i^0 = x_i^{\text{value}} + x_i^{\text{role}} \tag{1}$$

where x_i^{value} is the embedding for the i^{th} value and x_i^{role} is the embedding for the i^{th} role. At the query position, the value embedding of the [MASK] token indicates that this property is in query and the role embedding provides information about the type of query property. Different from existing transformer-based models [15, 8, 52], we do not use positional embeddings to indicate the position of each role-value pair in the n-ary query since, unlike natural language sentences or triples, there is no particular order for object properties. As the latter components of the model are permutation-invariant, removing the positional embeddings also allows our model to efficiently learn from object properties represented in n-ary observations.

B. Transformer Encoder

The transformer encoder takes the embedded input $\{h_1^0, ..., h_n^0\}$ and builds a contextualized representation $\{h_1^L, ..., h_n^L\}$ where L is the number of transformer layers in the transformer encoder. Each transformer layer applies the following transformation to the input:

$$\hat{g}^l = \text{MultiAttn}(h^{l-1}, h^{l-1}, h^{l-1}) \tag{2}$$

$$g^l = \text{LayerNorm}(\hat{g}^l + h^{l-1}) \tag{3}$$

$$\hat{h}^l = \text{FFN}(g^l) \tag{4}$$

$$h^l = \text{LayerNorm}(\hat{h}^l + g^l) \tag{5}$$

where MultiAttn is a multi-head self-attention mechanism, which we discuss in more depth below. FFN is a fully-connected feedforward network, which is applied to each position of the input separately and identically. The FFN consists of two linear fully-connected layers with a ReLU activation in between. Residual connections [24] are applied both after MultiAttn and FFN, which are followed by layer normalizations [4].

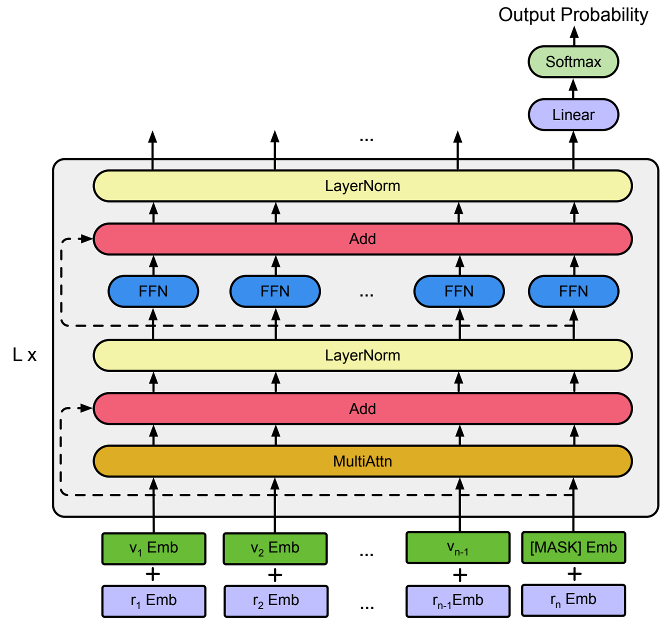

Fig. 3: Model architecture including the embedding layers, the transformer encoder, and a feed-forward layer for predicting probabilities of properties.

C. Multi-Head Attention

The core component of the transformer encoder is the multi-head attention mechanism, which builds on the scaled dot-product attention function. An attention function takes in a query and a set of key-value pairs. The output is computed as a weighted sum of the values, where the weight assigned to each value is based on the compatibility of the query with the corresponding key. The scaled dot-product attention performs the attention computation efficiently by computing on a set of queries simultaneous with matrix multiplication. The queries, keys, and values are stacked together into matrix $Q \in \mathbb{R}^{n_{\text{query}} \times d_{\text{model}}}$, $K \in \mathbb{R}^{n_{\text{key}} \times d_{\text{model}}}$, and $V \in \mathbb{R}^{n_{\text{value}} \times d_{\text{model}}}$, where n_{query} is the number of queries. Formally, the scaled dot-product attention is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{6}$$

where d_k is the dimension of queries and keys, and serves as a scaling factor for stabilizing gradients.

Instead of computing the attention function once, the multi-head attention has H heads, where each head performs a scaled dot-product attention. This allows each head to attend to different combinations of the input. In order to reason about the information at different representational space, Q, K, and V are also uniquely projected prior to the attention being computed. Specifically,

$$T_h = \text{Attention}(QW_h^Q, KW_h^K, VW_h^V) \tag{7}$$

where T_h is the output of a single attention head. $W_h^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_h^K \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_h^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ are learned linear projection weights for query, key, and value. In order to maintain the computation efficiency, d_k is chosen to be d_{model}/H. The outputs of the attention heads are concatenated and projected to form the final output of the multi-head attention:

$$\text{MultiAttn}(Q, K, V) = \text{Concat}(T_1, ..., T_H)W^O \tag{8}$$

where $W^O \in \mathbb{R}^{Hd_v \times d_{\text{model}}}$ is a learned output projection.

In our model, we use self-attention. Therefore, Q, K, and V are all constructed from h^{l-1}. Each position can freely attend to all positions in the input, thus aiding in modeling inter-relations between them.

D. Classification and Training

The final layer uses learned linear transformation and softmax function to convert the encoded input to predicted probabilities of properties. Specifically,

$$p_n = \text{softmax}(E_{\text{value}}\, \text{FCN}(h_n^L)) \tag{9}$$

where FCN is a fully connected layer and E_{value} is the learned embedding matrix used to create input value embeddings.

During training, we construct the masked input by replacing only a single value in an n-ary observation with the [MASK] token. We perform this procedure exhaustively for all values and all n-ary observations in the training set. We then group n-ary observations sharing the same masked instances and use their ground-truth values at the query position to construct a one-hot label (continuous-valued properties are discretized). Scoring multiple instances simultaneously is also known as the 1-N setting [14] and helps reduce training and inference time. We use cross-entropy between the ont-hot label and prediction as training loss. We use label smoothing [47] to prevent overfitting.

E. Implementation Details

All components of the model are trained end-to-end. The best set of parameters is found to be $L = 1$, $H = 4$, $d_{\text{model}} = 240$. We used Adam [26] for optimization. We implement our models using PyTorch and train on a Nvidia GTX1080Ti gpu.

VI. Experiments On LINK Dataset

In this section, we use the value prediction task to assess our model's ability to learn n-ary relations between object properties. In the value prediction task, the model is presented with a previously unseen n-ary observation, and must predict a single missing value given the value's role and all other role-value pairs in the instance.

A. Experimental Setup

Data Split: In our dataset, each n-ary observation corresponds to a situated object instance. To prevent test leakage, we first split object instances in the dataset into 70% training, 15% testing, and 15% validation. Situated object instances are then assigned to the correct set based on its corresponding object instance.

Metrics: For each missing value in a test instance, we obtain probabilities of candidate values from the model. Then the candidate values are sorted in descending order based on the probabilities. The rank of the ground-truth value v_n is used to compute metric scores. During ranking, we adopt the filtered setting [14] to remove any value v'_n different from v_n if $\{r_1 : v_1, ..., r_{n-1} : v_{n-1}, r_n : v'_n\}$ exists in the train, validation, or test set. This whole procedure is repeated for each value of each testing instance in the test set. We report standard metric Mean Reciprocal Rank (MRR) and proportion of ranks no larger than 1, 2, and 3 (Hits@1, 2, and 3). For both MRR and Hits, a higher score indicates better performance.

Baselines We compare against the following baselines:

- **Co-Occur** learns co-occurrence frequency of entities. This model has been used for modeling semantic relations in various robotic applications, including modeling object object co-occurrence [28, 56], object affordance co-occurrence [9], and object grasp co-occurrence [29]. We apply this model to learn the co-occurrence frequency of object class with object properties in our experiments.
- **TuckER** is a recent state of the art knowledge graph embedding model [5]. In this paper, we compare to two variants of TuckER. The regular TuckER model follows existing work [13, 3] to model binary relations between object class and other object properties.
- **TuckER+** is a TuckER embedding model we implement to model binary relations between all pairs of property types (e.g., color and material, shape and location); it approximates an n-ary relation with a combination of binary relations.
- **NaLP** is a neural network model developed for modeling n-ary relational data in knowledge graphs [21]. NaLP explicitly models the relatedness of all the role-value pairs in an n-ary observation. We apply this model to learn n-ary relations between object properties.
- **Markov Logic Network (MLN)** represents probabilistic logic languages that have been used to model complex semantic relations in various robotic domains [41, 58, 42, 2, 11]. We closely follow prior work to specify probabilistic rules for our domain.

B. Results

As shown in Table III, our model outperforms existing methods by significant margins on all metrics. Compared to the second-best model, **MLN**, our model achieves a 10% increase in MRR while reducing training and testing time by 166 times. In comparison with **NaLP**, another model developed specifically for modeling n-ary data, our model's superior performance confirms that the transformer structure and multi-head attention mechanism are more effective at learning the complex semantic relations between object properties. We also observe that **TuckER+**, which learns binary relations between all pairs of object properties, outperforms the regular **TuckER**. This result demonstrates that modeling only class-level semantic knowledge can lead to over-generalization, and that reasoning about the differences between object instances is crucial. It is worth noting that **NaLP** and **TuckER** variants are not able to outperform the simpler **Co-Occur** model. **TuckER** variants are good at learning latent representation of the global

TABLE III: Results% of our model and baseline models.

Model	Metric Scores				Time (min)	
	MRR	Hits@1	Hits@2	Hits@3	Training	Testing
Co-Occur	63.0	44.3	67.3	80.2	<1	3
TuckER	58.7	38.5	59.9	79.3	<1	3
TuckER+	62.5	43.0	65.9	81.4	2	3
NaLP	57.9	38.9	60.3	75.8	8	10
MLN	66.1	50.2	68.8	82.0	420	487
Transformer (Ours)	**76.3**	**63.3**	**79.4**	**89.1**	3	3

TABLE IV: Ablation on input encoder design

Embeddings			Metric Scores			
V	R	Pos	MRR	Hits@1	Hits@2	Hits@3
✓	✓		**76.3**	**63.3**	**79.4**	**89.1**
✓			75.3	62.3	79.0	86.8
✓	✓	✓	74.0	59.9	77.7	87.5
✓		✓	74.0	59.9	77.7	87.5

TABLE V: MRR% of our model and baseline models for each property type.

	Class	Mat	Color	Trans	Dim	Phys	Shape	Temp	Full	Damp	Clean	Room	Loc	Price	Weight	Size
# Values	11	8	15	3	10	6	9	3	3	3	3	11	117	3	3	3
Random	27.1	33.3	19.9	60.8	18.7	37.7	22.6	61.4	61.5	61.5	61.0	28.5	4.7	60.2	60.7	60.5
Co-Occur	/	55.5	39.2	83.1	17.9	76.1	64.4	91.0	77.4	74.5	67.1	56.1	44.3	56.9	70.7	**70.9**
TuckER	/	56.1	37.7	84.0	16.7	74.3	57.8	83.9	59.9	71.6	61.4	57.3	31.4	55.6	64.6	69.0
TuckER+	53.7	60.0	42.2	85.1	20.1	74.8	60.4	90.9	69.7	72.4	62.6	60.9	39.2	59.7	73.8	65.1
NaLP	45.0	47.4	39.0	84.4	17.7	69.2	52.4	91.0	68.5	70.8	67.7	55.2	23.1	59.0	61.6	61.4
MLN	62.9	**79.1**	72.5	95.7	25.3	79.4	64.3	90.0	69.1	69.5	65.3	67.7	4.6	68.9	75.8	63.7
Transformer (ours)	**72.3**	78.9	**73.2**	**97.1**	**43.8**	**84.1**	**75.7**	**92.3**	**90.7**	**82.2**	**90.2**	**68.5**	**59.6**	**74.4**	**76.6**	61.8

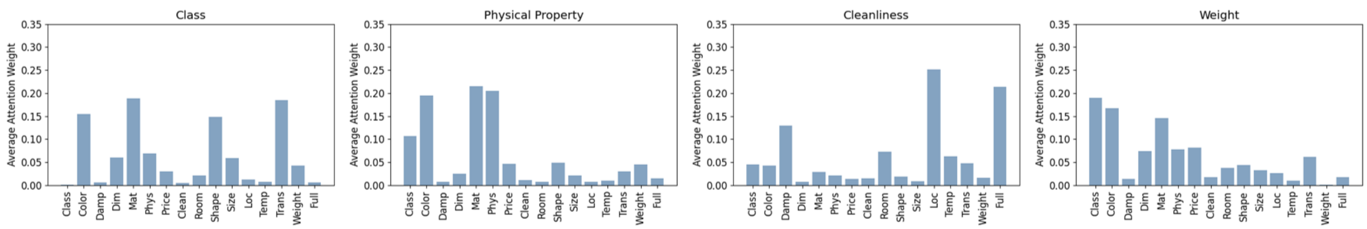

Fig. 4: Visualizations of attention weights illustrate that different amount of information from each property type is used by our model to predict different types of properties.

structure, but our analysis shows that they do not capture the frequency of the binary relations well. **NaLP** has shown to be effective at modeling n-ary facts mainly on dataset with 2 to 6 role-values, but it struggles to learn n-ary relations in our data which can have up to 24 role-values.

Further analyzing MRR for each type of query shown in Table V, we see that our model outperforms existing models in predicting most of the properties. We also notice that the baselines have degraded performance at predicting property types with many candidate values (e.g., location, room, and dimension). **MLN** especially struggle to predict the location role with 117 possible values. One potential explanation is the closed world assumption being made by **MLN**. Our model uses label smoothing to prevent being overconfident at negative training examples and has demonstrated good performance even for these many-valued role types.

C. Ablation on Input Encoder Design

We investigate our input encoder design with an ablation study. Specifically, we examine the effect of the role embeddings and positional embeddings (discussed in Section V-A). Results in Table IV show that enforcing the order of role-value pairs in an n-ary observation using the positional embeddings results in a drop in performance. The results also confirm that role embedding is useful for modeling n-ary relations represented as role-value pairs.

D. Visualizing Attention

To understand why our transformer-based model is effective at modeling n-ary relational data, we visualize the multi-head attention weights, i.e., softmax$(\frac{QK^T}{\sqrt{d_k}})$. Figure 4 shows the average attention weight assigned to each role when predicting class, physical property, cleanliness, and weight. The attention mechanism exhibits n-ary relational reasoning patterns, which also correspond strongly with human intuition—for example, dampness, location, and content of an object aids in predicting its cleanliness. Baseline models cannot perform this type of reasoning and thus are not able to model object properties as well as our model.

VII. Robot Experiment: Object Search

In this section, we demonstrate how our model can be used to enable a household robot to effectively locate objects. Our experiment serves two purposes, i) to validate our model in a realistic physical setting with non-AMT users, and ii) to test our model's ability to handle queries that reflect realistic use cases, such as a human asking a robot to find a cold beverage or collect dirty dishes. Queries used in this study utilize only a sparse set of known properties[2], and the robot's task is to predict multiple unknown properties. Specifically, we seek to predict the room and location of each object.

We set up a home environment in our lab with 4 rooms and typical household furniture (Figure 5). We also generated

[2]Human users are unlikely to phrase requests with long adjective sequences.

TABLE VI: Results% on object search

	Hits@1	Hits@2	Hits@3	Hits_Any@1	Hits_Any@2	Hits_Any@3
Human Baseline	34.8 ± 6.5	52.0 ± 7.0	64.7 ± 7.3	64.7 ± 7.2	83.2 ± 7.0	90.6 ± 3.9
Co-Occur	**20.0**	29.0	36.8	**49.0**	68.0	80.0
TuckER+	8.4	20.0	29.6	37.2	60.0	74.4
NaLP	1.2	2.8	4.4	7.2	12.4	16.8
Transformer	17.2	**34.4**	**48.0**	42.4	**73.6**	**87.6**

Fig. 5: Home environment for object search experiment.

Fig. 6: 3D floor plan for remotely collecting preferred locations of objects from 5 users.

a corresponding 3D floor plan of the environment (adding an additional bathroom), which listed 24 possible locations for storing objects (Figure 6). We then recruited 5 users, and had them label their preferred location for 50 object instances sampled from our dataset. For each object, the user was shown an image of the object, given 1-3 properties describing the state of the object (i.e., cleanliness, temperature, dampness, and content), and then asked to list 3 ranked likely locations for the object.

We compared the performance of our model against **NaLP**, **Co-Occur**, and **TuckER+**. We left **MLN** out because of its exceedingly long inference time on queries with partial evidence (as a large number of properties other than the query properties were missing). All models were trained on our complete dataset to validate against collected user data. All models had access to the properties given to the human users as well as the class and material of the object. To predict likely room-location combinations, separately predicted probabilities of the two properties were multiplied and ranked.

We use Hits@K and Hits_Any@K as metrics. Hits@1,2,3 indicate the percentage of times that a model correctly predicted a user's most preferred location of an object within 1, 2, and 3 attempts, respectively. We also introduce Hits_Any@K, which considers a prediction correct if it matches any one of the 3 locations listed by a user, without rank order.

Table VI summarizes the result of this experiment. We also report the human baseline, which we compute by cross-validating each user against the other users. We observe that only our model is able to reach within the range of human performance at Hits_Any@2 and 3. **Co-Occur** outperforms our model at Hits@1 and Hits_Any@1, suggesting that class-level frequency is a good heuristic for finding objects if given only one chance. However, given that only approximately 30% of objects can be retrieved in one shot, even by humans, we argue that our model is more beneficial in the general use case.

Beyond quantitative difference between our model and baselines, we also demonstrate the qualitative improvement on a Fetch robot [54]. The robot is equipped with the navigation stack developed in [6] for mapping and navigation, and the method introduced in [32] for object detection and grasping. As shown in Figure 7, the difference (A, B) between our model and **Co-Occur** is clear as our model takes into account of the properties of objects (e.g., cold, dry, clean) while **Co-Occur** searches the same locations for different cups. We also show in Figure 8 that our model is able to find objects considering both immutable (material in E and F) and mutable properties of objects (dampness in C and D).

VIII. Robot Experiment: Integrating with Multimodal Perception

In this section, we examine whether a robot with limited sensing ability can use our model to infer object properties that cannot be directly observed. This experiment also tests whether our model can generalize learned n-ary knowledge to new object instances in the real world.

In this experiment, a robot is tasked to predict either an unknown immutable property of an object based on its class, color, material, and room, or to predict an unknown mutable property based on class, color, material, room, temperature, and location. We use the same Fetch robot, object detection, and mapping as the previous experiment. Material is detected by the robot using a spectrometer, the SCiO sensor [16]. Temperature is detected using a Melexis contact-less infrared sensor connected to an Arduino microcontroller. Color is

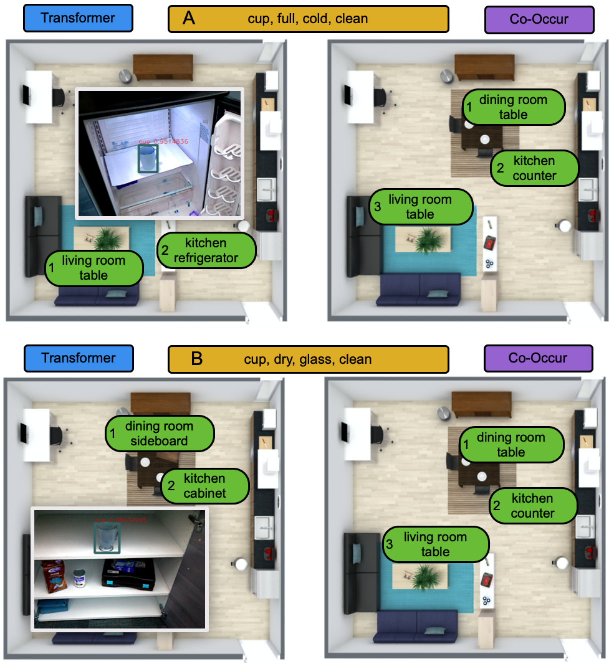

Fig. 7: Two object search tests comparing our model with Co-Occur. Provided information is shown on top.

Fig. 8: Our model searches for different places based on given object properties.

Fig. 9: The Fetch robot uses infrared temperature sensor and spectroscopy (bottom left) to detect the temperature and material of each novel object (top left). Our model leverages extracted information (shown on top of each figure on the right) to predict an unknown object property (shown on bottom).

detected using OpenCV. As shown in Figure 9, we test on 22 objects which are semantically different from objects in our dataset (e.g., no ceramic pan and plastic box exist in our dataset).

In this experiment, our model is able to correctly predict 34/52 (65%) of the queried object properties. In comparison, the second best performing models, **TuckER+** and **Co-Occur**, both correctly predict 24/52 (46%). The material classification algorithm correctly detects object material 45/52 (87%) times. Figure 9 shows examples of the queries.

IX. Conclusion

This work addresses the problem of predicting semantic properties of objects based on partial observations. Our approach uses a contextualized language model to efficiently model and learn n-ary relations between object properties. We contribute LINK, a dataset containing the most diverse set of properties to date. Evaluation of our model shows significant improvements over prior methods. The object search and perception experiments also demonstrate the effectiveness of our method on a Fetch robot.

References

[1] R Abboud, II Ceylan, T Lukasiewicz, and T Salvatori. Boxe: A box embedding model for knowledge base completion. *NeurIPS Proceedings*, 33, 2020.

[2] Paola Ardón, Èric Pairet, Ronald PA Petrick, Subramanian Ramamoorthy, and Katrin S Lohan. Learning grasp affordance reasoning through semantic relations. *IEEE Robotics and Automation Letters*, 4(4):4571–4578, 2019.

[3] Jacob Arkin, Daehyung Park, Subhro Roy, Matthew R Walter, Nicholas Roy, Thomas M Howard, and Rohan Paul. Multimodal estimation and communication of latent semantic knowledge for robust execution of robot instructions. *The International Journal of Robotics Research*, page 0278364920917755, 2020.

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[5] Ivana Balazevic, Carl Allen, and Timothy Hospedales. TuckER: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5185–5194, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1522. URL https://www.aclweb.org/anthology/D19-1522.

[6] Siddhartha Banerjee, Angel Daruna, David Kent, Weiyu Liu, Jonathan Balloch, Abhinav Jain, Akshay Krishnan, Muhammad Asif Rana, Harish Ravichandar, Binit Shah, Nithin Shrivatsav, and Sonia Chernova. Taking recoveries to task: Recovery-driven development for recipe-based robot tasks. In *International Symposium on Robotics Research (ISRR)*, 2019. URL http://www.banerjs.com/pdf/banerjs_isrr2019.pdf.

[7] Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

[8] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, 2019.

[9] Yu-Wei Chao, Zhan Wang, Rada Mihalcea, and Jia Deng. Mining semantic affordances of visual object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4259–4267, 2015.

[10] Haonan Chen, Hao Tan, Alan Kuntz, Mohit Bansal, and Ron Alterovitz. Enabling robots to understand incomplete natural language instructions using commonsense reasoning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1963–1969. IEEE, 2020.

[11] Sonia Chernova, Vivian Chu, Angel Daruna, Haley Garrison, Meera Hahn, Priyanka Khante, Weiyu Liu, and Andrea Thomaz. Situated bayesian reasoning framework for robots operating in diverse everyday environments. In *International Symposium on Robotics Research (ISRR)*, 2017.

[12] Vivian Chu, Ian McMahon, Lorenzo Riano, Craig G McDonald, Qin He, Jorge Martinez Perez-Tejada, Michael Arrigo, Trevor Darrell, and Katherine J Kuchenbecker. Robotic learning of haptic adjectives through physical interaction. *Robotics and Autonomous Systems*, 63:279–292, 2015.

[13] Angel Daruna, Weiyu Liu, Zsolt Kira, and Sonia Chetnova. Robocse: Robot common sense embedding. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9777–9783. IEEE, 2019.

[14] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[16] Zackory Erickson, Nathan Luskey, Sonia Chernova, and Charles C Kemp. Classification of household materials via spectroscopy. *IEEE Robotics and Automation Letters*, 4(2):700–707, 2019.

[17] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1778–1785. IEEE, 2009.

[18] Vittorio Ferrari and Andrew Zisserman. Learning visual attributes. *Advances in neural information processing systems*, 20:433–440, 2007.

[19] Jeremy A Fishel and Gerald E Loeb. Bayesian exploration for intelligent identification of textures. *Frontiers in neurorobotics*, 6:4, 2012.

[20] Mikhail Galkin, Priyansh Trivedi, Gaurav Maheshwari, Ricardo Usbeck, and Jens Lehmann. Message passing for hyper-relational knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7346–7359, 2020.

[21] Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. Link prediction on n-ary relational data. In *The World Wide Web Conference*, pages 583–593, 2019.

[22] Martin Günther, JR Ruiz-Sarmiento, Cipriano Galindo, Javier Gonzalez-Jimenez, and Joachim Hertzberg. Context-aware 3d object anchoring for mobile robots. *Robotics and Autonomous Systems*, 110:12–32, 2018.

[23] Birgit Hamp and Helmut Feldweg. Germanet-a lexical-semantic net for german. In *Automatic information extraction and building of lexical semantic resources for*

NLP applications, 1997.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[25] Patrick Jenkins, Rishabh Sachdeva, Gaoussou Youssouf Kebe, Padraig Higgins, Kasra Darvish, Edward Raff, Don Engel, John Winder, Francisco Ferraro, and Cynthia Matuszek. Presentation and analysis of a multimodal dataset for grounded language learning. *arXiv preprint arXiv:2007.14987*, 2020.

[26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

[27] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[28] Lars Kunze, Chris Burbridge, Marina Alberti, Akshaya Thippur, John Folkesson, Patric Jensfelt, and Nick Hawes. Combining top-down spatial reasoning and bottom-up object class recognition for scene understanding. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2910–2915. IEEE, 2014.

[29] Safoura Rezapour Lakani, Antonio J Rodríguez-Sánchez, and Justus Piater. Exercising affordances of objects: A part-based approach. *IEEE Robotics and Automation Letters*, 3(4):3465–3472, 2018.

[30] Séverin Lemaignan, Mathieu Warnier, E Akin Sisbot, Aurélie Clodic, and Rachid Alami. Artificial cognition for social human–robot interaction: An implementation. *Artificial Intelligence*, 247:45–69, 2017.

[31] Gi Hyun Lim, Il Hong Suh, and Hyowon Suh. Ontology-based unified robot knowledge for service robots in indoor environments. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41 (3):492–509, 2011.

[32] Weiyu Liu, Angel Daruna, and Sonia Chernova. Cage: Context-aware grasping engine. In *International Conference on Robotics and Automation (ICRA)*, 2020. URL https://arxiv.org/abs/1909.11142.

[33] Yu Liu, Quanming Yao, and Yong Li. Generalizing tensor decomposition for n-ary relational knowledge bases. In *Proceedings of The Web Conference 2020*, pages 1104–1114, 2020.

[34] Shan Luo, Wenzhen Yuan, Edward Adelson, Anthony G Cohn, and Raul Fuentes. Vitac: Feature sharing between vision and tactile sensing for cloth texture recognition. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2722–2727. IEEE, 2018.

[35] Dermot Lynott and Louise Connell. Modality exclusivity norms for 423 object properties. *Behavior Research Methods*, 41(2):558–564, 2009.

[36] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *International Conference on Learning Representations*, 2018.

[37] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[38] Bogdan Moldovan and Luc De Raedt. Occluded object search by relational affordances. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 169–174, 2014.

[39] Michal Nazarczuk and Krystian Mikolajczyk. Shop-vrb: A visual reasoning benchmark for object perception. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6898–6904. IEEE, 2020.

[40] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2015.

[41] Daniel Nyga, Ferenc Balint-Benczedi, and Michael Beetz. Pr2 looking at things—ensemble learning for unstructured information processing with markov logic networks. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3916–3923. IEEE, 2014.

[42] Daniel Nyga, Subhro Roy, Rohan Paul, Daehyung Park, Mihai Pomarlan, Michael Beetz, and Nicholas Roy. Grounding robot plans from natural language instructions with incomplete world knowledge. In *Conference on Robot Learning*, pages 714–723, 2018.

[43] Paolo Rosso, Dingqi Yang, and Philippe Cudré-Mauroux. Beyond triplets: hyper-relational knowledge graph embedding for link prediction. In *Proceedings of The Web Conference 2020*, pages 1885–1896, 2020.

[44] Ashutosh Saxena, Ashesh Jain, Ozan Sener, Aditya Jami, Dipendra K Misra, and Hema S Koppula. Robobrain: Large-scale knowledge engine for robots. *arXiv preprint arXiv:1412.0691*, 2014.

[45] Mohit Shridhar and David Hsu. Interactive visual grounding of referring expressions for human-robot interaction. In *Proceedings of Robotics: Science and Systems*, 2018.

[46] Yuyin Sun, Liefeng Bo, and Dieter Fox. Attribute based object identification. In *2013 IEEE international conference on robotics and automation*, pages 2096–2103. IEEE, 2013.

[47] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[48] Moritz Tenorth and Michael Beetz. Representations for robot knowledge in the knowrob framework. *Artificial Intelligence*, 247:151–169, 2017.

[49] Jesse Thomason, Jivko Sinapov, Raymond J Mooney, and

Peter Stone. Guiding exploratory behaviors for multi-modal grounding of linguistic descriptions. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[50] Karthik Mahesh Varadarajan and Markus Vincze. Afnet: The affordance network. In Kyoung Mu Lee, Yasuyuki Matsushita, James M. Rehg, and Zhanyi Hu, editors, *Computer Vision – ACCV 2012*, pages 512–523, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-37331-2.

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[52] Quan Wang, Pingping Huang, Haifeng Wang, Songtai Dai, Wenbin Jiang, Jing Liu, Yajuan Lyu, Yong Zhu, and Hua Wu. Coke: Contextualized knowledge graph embedding. *arXiv preprint arXiv:1911.02168*, 2019.

[53] Jianfeng Wen, Jianxin Li, Yongyi Mao, Shini Chen, and Richong Zhang. On the representation and embedding of knowledge bases beyond binary relations. In *IJCAI*, 2016.

[54] Melonee Wise, Michael Ferguson, Derek King, Eric Diehr, and David Dymesich. Fetch and freight: Standard platforms for service robot applications. In *Workshop on autonomous mobile service robots*, 2016.

[55] Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. In *Proceedings of Seventh International Conference on Learning Representations (ICLR 2019)*, May 2019.

[56] Zheng Zeng, Adrian Röfer, Shiyang Lu, and Odest Chadwicke Jenkins. Generalized object permanence for object retrieval through semantic linking maps. In *IEEE ICRA 2019 Workshop on High Accuracy Mobile Manipulation in Challenging Environments*, 2019.

[57] Richong Zhang, Junpeng Li, Jiajie Mei, and Yongyi Mao. Scalable instance reconstruction in knowledge bases via relatedness affiliated embedding. In *Proceedings of the 2018 World Wide Web Conference*, pages 1185–1194, 2018.

[58] Yuke Zhu, Alireza Fathi, and Li Fei-Fei. Reasoning about object affordances in a knowledge base representation. In *European conference on computer vision*, pages 408–424. Springer, 2014.