Functionality Discovery and Prediction of Physical Objects

Lei Ji[1,2,3], Botian Shi[4], Xianglin Guo[4], Xilin Chen[1,2]
[1]Institute of Computing Technology, CAS, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
[3]Microsoft Research Asia, Beijing, China
[4]Beijing Institute of Technology, Beijing, China
leiji@microsoft.com, (botianshi, guoxianglin)@bit.edu.cn, xlchen@ict.ac.cn

Abstract

Functionality is a fundamental attribute of an object which indicates the capability to be used to perform specific actions. It is critical to empower robots the functionality knowledge in discovering appropriate objects for a task e.g. *cut cake* using *knife*. Existing research works have focused on understanding object functionality through human-object-interaction from extensively annotated image or video data and are hard to scale up. In this paper, we (1) mine object-functionality knowledge through pattern-based and model-based methods from text, (2) introduce a novel task on physical object-functionality prediction, which consumes an image and an action query to predict whether the object in the image can perform the action, and (3) propose a method to leverage the mined functionality knowledge for the new task. Our experimental results show the effectiveness of our methods.

1 Introduction

Functionality indicates the capability of an object to be used to perform specific actions. According to the study of psychologist (Gibson 2014; Oakes and Madole 2008), functionality is a fundamental attribute for an object to be perceived by human, which is as important as appearance for object recognition. Learning object functionality is significantly important for robots to interact with environment, which has been widely studied in robotic community.

Figure 1: Object-functionality prediction task. Given the query <*cut*, *cake*>, whether the object in the image can be used as tool to *cut cake*. The *knife* and *spoon* should output *yes*, and others should output *no*.

With the recent advances in NLP tasks, including question answering(Petrochuk and Zettlemoyer 2018) and dialogue (Serban et al. 2016; Young et al. 2018), chat with robots has achieved significant progress. Besides, in robotic community, interacting with robot to complete tasks is a long-standing research problem, which required both language understanding and physical grounding. Generally, human would like to give orders like *cut cake* without specifying the tool e.g. ***cut*** *the cake using* ***knife***. The robot should be able to find a tool autonomously, which can be potentially used to *cut cake*.

According to our study, WikiHow[1] contains sentences of guiding human to perform physical actions. Within 7,800,973 sentences we have collected, around 4% sentences mention the actual tool explicitly. After the investigation on 20 commonly used actions which require tool to perform, about 6% sentences explicitly mention the tool. Human regards the object functionality as commonsense knowledge and would like to not specify the tool. Without explicitly mentioned, it is critical to empower robots the functionality knowledge in discovering appropriate objects for a task. We focus on two research problems:

1. What *functionalities/actions* an *object* has?
2. Whether the *object* can perform the *action* on another *object*?

Existing knowledge graph ConceptNet(Speer and Havasi 2012) has *UsedFor* relation, which were collected in a crowdsourcing way and far from satisfying application demands due to low coverage. Previous research works have mainly focused on understanding object functionality through human-object-interaction from extensive annotated image or video data and are hard to scale up. According to our study on Visual Genome(Krishna et al. 2017), which is an extensive labeled scene graph of each image, most relations are spatial (e.g. on, near, etc.) relations and only a few are physical (e.g. wear, hold, etc.) relations.

In this paper, we first mine the object-functionality knowledge through pattern-based and model-based methods from large amount of text data, which improve the coverage to a great extent. The extracted object-functionality are represented as knowledge tuples <head, action, tail>, which indicates the *head object* performs an *action* on the *tail object*.

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

[1]https://www.wikihow.com/Main-Page

In real applications, a robot seeks visible objects that can be used to complete the task based on appearance. Motivated by this, we introduce a new task and dataset on physical object-functionality prediction, which consumes an object image and an action query <action, object> to predict whether the object in the image can perform the action. Figure 1 gives a showcase of the task. As an example, we can use *knife* or *spoon* to *cut the cake* but not *scissors*, *hammer* or *rolling pin*. To resolve this task, we propose a novel model to predict the object-functionality through both visual appearance and the extracted functionality knowledge. More specifically, on one hand, we use a CNN based model to perform object and functionality classification. On the other hand, we directly use the predicted object and the action query to construct a tuple <head, action, tail> and employ both PRA (Path ranking algorithm)(Lao, Mitchell, and Cohen 2011) and Dismult(Yang et al. 2015) module for inference. Then we finalize the score through a linear combination.

We conduct experiments on evaluating the quality and coverage of the mined object-functionality knowledge. Experimental results of applying the knowledge for object-functionality prediction task show the effectiveness of our model. To sum up, the contributions of this paper are:

1. We propose to mine functionality from large scale text data.
2. We introduce a new task for object-functionality prediction and constructing a new dataset.
3. We design a model through visual-based recognition and knowledge inference modules to resolve the task.

2 Related Works

Object affordance and functionality have been studied for years and have attracted more attention in computer vision and robotics community. According to Gibson(Gibson 2014), Object affordance reveals the possible actions the object can perform or be performed on them when interacting with environment. In this work, we focus on object functionality, which is the action that object can perform. Yao(Yao, Ma, and Fei-Fei 2013) discovered object functionality through human interaction with the object. Zhu(Zhu, Fathi, and Fei-Fei 2014) proposed to infer object affordance by reasoning in knowledge base which consists of visual, physical, categorical as well as HOI (human-object-interaction) concepts. Zhu(Zhu, Zhao, and Chun Zhu 2015) introduced a method to understand functionality by imaging actions on physical concepts. However, it is hard to collect large scale HOI dataset with annotation. Azuma(Azuma, Takiguchi, and Ariki) shown that functionality can be effectively recognized from appearance according to attributes like shape or material. Different from previous work, we first mine the functionality knowledge and predict the functionality from both object appearance and knowledge,

Knowledge extraction aims to mine propositions in the form of <subject, verb, object > from large scale corpus. NELL (Never Ending Language Learner)(Mitchell et al. 2018) extracted structured knowledge by bootstrapping techniques. Recently, neural based open information extraction(Stanovsky et al. 2018) has been widely used to mine knowledge. In this paper, we adopt RnnOIE(Stanovsky et al. 2018) which is a Bi-LSTM model for sequence tagging. Object functionality is rarely declared as the form of subject-verb-object in the corpus and we apply several post processing methods for further extraction.

Commonsense knowledge ConceptNet(Speer and Havasi 2012) is the most widely used commonsense knowledge graph and is collected in a crowdsourcing way. WebChild(Tandon et al. 2014) extracted commonsense knowledge from text automatically. Recent advances attempted to extract knowledge through visual images. NEIL(Chen, Shrivastava, and Gupta 2013) is a pipeline to extract knowledge from images endlessly. Xin(Lin and Parikh 2015; Vedantam et al. 2015) has proposed to extract commonsense knowledge by abstract images. Yatskar(Yatskar, Ordonez, and Farhadi 2016) extract visual commonsense knowledge mainly on spatial relations such as *on, above, besides, touch*, etc. However, these knowledge graphs neglect physical actions. Chao(Chao et al. 2015) has proposed to mine semantic affordance through text mining, visual mining and collaborative filtering methods. We not only mine functionality, but also incorporate the knowledge for further object-functionality prediction task.

Incorporating knowledge Tremendous research works aim to incorporate knowledge for AI agents e.g. KBQA(Petrochuk and Zettlemoyer 2018), Dialogue(Young et al. 2018), VQA(Lu et al. 2018). Knowledge embedding are widely used techniques for link prediction and incorporating knowledge in many tasks. which embed discrete tokens to vector representation, such as TransE(Bordes et al. 2013), Dismult(Yang et al. 2015), URGE(Chen et al. 2019). In this work, we adopt Dismult to predict the relationship between the head and tail entities. Different from knowledge embedding, we also predict object and its functionality through PRA(Lao, Mitchell, and Cohen 2011) with regard to: (1) PRA(Lao, Mitchell, and Cohen 2011) uses the explicit pathes to predict relations (2) Existing knowledge embedding models aim to encode encyclopedic knowledge instead of commonsense knowledge. Experimental results have shown that combining Dismult and PRA can effectively predict object functionality.

3 Object-Functionality Prediction Task

In this section, we first formulate the object-functionality prediction task, then discuss the framework, and finally introduce the dataset.

Problem Formalization

Given an image of object I, and an action query Q as <action, object>, predict whether the object I can be used to complete the task Q. This is formalized as a classification problem.

Framework

We discuss the framework to perform the task. Firstly, we extract the object-functionality knowledge from various text dataset; Secondly, we consolidate these candidates into one knowledge graph; Finally, we propose a model to predict the plausibility score through object and functionality classification module followed by an inference module. Figure 2 shows the overview of the framework.

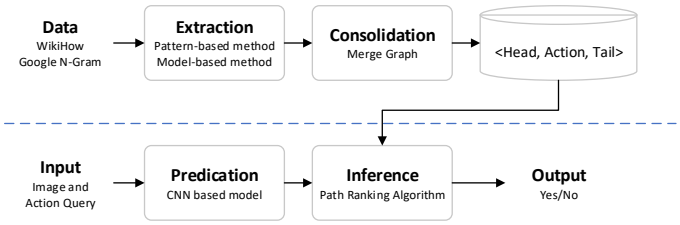

Figure 2: Framework

Data Collection

We collect a dataset for the object-functionality prediction task. This dataset contains <head (image), action, tail> with a label 1(yes) or 0(no). We start by selecting actions which require tools to complete. Then we label (1) the <head, action> pairs to judge whether the head object can execute the action; (2) the <head, action, tail> tuples to decide whether the head object can perform the action on the tail. Finally, we crawl and clean the images of objects.

Actions Following (Chao et al. 2015), We first extract the 100 most frequently used and physically visible actions from the dataset in (Chao et al. 2015). Chao(Chao et al. 2015) labeled the visualness score of verbs, and we use their label and select the visible verbs (score >3.6). We also removed several vague actions like *have*, *make*, *be* and actions which do not require a *tool* to perform such as *think*, *run*. At last, we select 20 actions listed in Table 1.

Table 1: Action List

break	cut	heat	roll
build	decorate	hit	shoot
clean	draw	kill	tie
contain	drink	paint	wash
cook	eat	repair	write

Head-Action pairs We select top 70 objects which are frequently used to perform the 20 actions from ConceptNet. Then we conduct Cartesian product on all 70 head objects and 20 actions into 1400 <head, action > pairs as candidates. After that, we ask two labelers to annotate the question:

Q: "whether the *object* can perform *action* ?"

Two labelers have the consistent labels for over 80% pairs and we asked another labeler to make a final judgment for the inconsistent labels. In total, we get 330 positive pairs and 1,070 negative pairs.

Action-Tail affordance pairs (Chao et al. 2015) has studied the affordance whether human can perform the actions on the objects e.g. *cut* is not affordable for *water* but is affordable for *cake*. We sample 200 action-tail pairs from all valid affordance dataset to avoid meaningless tuples.

Head-Action-Tail tuples We merged the 330 <head, action> and 200 <action, tail> reasonable pairs by a inner join on action, and get 3,900 tuples. Then we ask labelers to annotate:

Q: "whether the *head object* can perform *action* on the *tail object*".

Similarly, two labelers are asked to label and one for confirmation. Altogether, we get 2,232 positive tuples. Table 2 shows some examples.

Table 2: <head, action > pair and <head, action, tail > tuple examples

Head	Action	Tail	Label
toothbrush	wash	-	1
knife	kill	-	1
ax	drink	-	0
bag	cut	-	0
toothbrush	wash	face	0
knife	clean	fish	1
ax	cut	tree	1
bag	contain	water	0

Figure 3 depicts the positive ratio for each action. From the result, we can see 1) all tuples with action *hit* are positive, which shows that this functionality is general and can perform on all objects in our dataset. 2) While for actions like *cook* and *roll*, there are more negative tuples than positive, which shows that the object can only perform the functionality to a small group of objects. Take *clean* as an example, *mop* can be used to clean *floor* but not *food*.

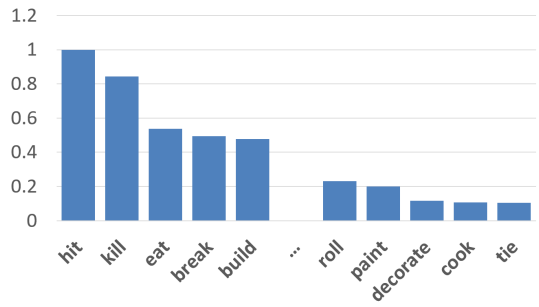

Figure 3: Positive ratio for each action

To collect the images, we crawl the top 200 images for each object from a commercial search engine. Then we clean

Table 3: List of patterns

	Type	Pattern
#1	Token	[*head* used for *doing tail*]
	Dependence tree	[nsubj(pass) root prep comp, dobj]
	POS tagging	[NN VB IN VB NN]
#2	Token	[*do tail* using *head*]
	Dependence tree	[root dobj comp dobj]
	POS tagging	[VB NN VB NN]
#3	Token	[*do tail* with *head*]
	Dependence tree	[root dobj prep pobj]
	POS tagging	[VB NN IN NN]

and collect 11,666 images in all. The overall dataset contains:

1. Head-Action pairs: 330 over 1,400 pairs are positive, which are used for evaluating quality of mined knowledge and functionality classification.
2. Head-Action-Tail tuples: 3,900 tuples to evaluate inference module.
3. Head(image)-Action-Tail tuples: 678,900 tuples in all used to evaluate end-to-end object-functionality prediction task.

4 Extract Object-Functionality Knowledge

In this section, we study the first research problem: What *functionality*/*actions* an *object* has? We employ both pattern-based and model-based knowledge mining methods to mine <head, action, tail> tuples and then evaluate the consolidated tuples.

UsedFor tuples in ConceptNet

ConceptNet(Speer and Havasi 2012) is a commonsense knowledge graph and has *UsedFor* relation indicating object functionality. We first select this relation, and extract the verb and object from the tail. For example, the tuple <knife, UsedFor, cutting cakes> is refined to <knife, cut, cake>.

Pattern-based method

We adopt a pattern-based method on Google Syntactic N-gram dataset (Lin et al. 2012), which parsed the dependency syntactic from large scale corpus. We directly apply patterns listed in Table 3 to extract tuples.

Model-based method

WikiHow is "*the world's most popular how-to website illustrating instructions for everything*", which contains sentences stating how to perform physical actions. In this paper, we collect 7,800,973 sentences to extract tuples.

We adopt RnnOIE(Stanovsky et al. 2018), a bi-LSTM sequence tagging model, to extract tuples. This model iteratively lends itself to BIO (Beginning-Inside-Outside) tagging to capture a wide range of propositions. The label types are predicate(V), argument(ARG) or others(O) with BIO scheme. Given the sentence, sequentially predict each token with one of the labels. RnnOIE generates multiple extractions from a single sentence through certain syntactic construction.

Table 4: A showcase of model based example

Cut the cake using a sharp knife and put one piece into a place
RnnOIE: **cut**: [V: Cut] [ARG1: the cake using a sharp knife and put one piece into a place] **using**: Cut [ARG0: the cake] [V: using] [ARG1: a sharp knife] and put one piece into a place **put**: [ARG0: Cut] [ARG0: the cake] using a sharp knife and [V: put] [ARG1: one piece] [ARG2: into a place]
Tuples: <, cut, the cake using...>, <the cake, using, a sharp knife>, <cut, put, one piece>, <the cake, put, one piece>
Our: <knife, cut, cake>
New label: Cut_{V} the_{o} $\text{cake}_{\text{ARG1}}$ using_{o} a_{o} sharp_{o} $\text{knife}_{\text{ARG0}}$ and_{o} put_{o} one_{o} piece_{o} into_{o} a_{o} place_{o}.

Table 4 lists one showcase of creating training data from existing RnnOIE results. Although the existing model can extract many tuples, none of them are object-functionality tuples. We apply several post processing rules to further extract the valid tuples. One sentence has multiple outputs. The first tuple <, cut, the cake using...> is taking *cut* as V and *using* clause as ARG1 and the second tuple <the cake, using, a sharp knife> is taking *using* as V and *a sharp knife* as ARG1. (1) we take V and ARG1 in the first tuple as action and tail object, and ARG1 in the second tuple as head object. Through this way, we get <a sharp knife, cut, cake using...>. (2) we apply dependency tree parser to ARG1 sentences in both tuple and take the root as head and tail objects respectively. Finally, we get the expected tuple <knife, cut, cake>. Similarly, for the *with* clause, the extraction has a *with* clause as ARG2, and we take the V as action, root of ARG1 as tail, and root of ARG2 as head object.

After post processing, we have a group of seed tuples and use them to relabel the sentences to train a model for extraction. The goal is to train a new RnnOIE model to directly predict object-functionality tuple. Table 4 shows an example of the label. In all, we have collected 266,827 sentences for training.

Implementation setting We use the RnnOIE open source code[2] to run the model using the PyTorch framework. The bi-LSTM model has 1 layer and each LSTM cell uses 128 hidden units followed by a ReLU activation function. We train the model for 500 epochs with mini batch size as 80 and 0.001 as learning rate. We use GloVe 100-dimensions word embeddings. Simultaneously, we use a dependency parser[3](Qi et al. 2018) to predict syntactic feature for extracting root entity.

[2]https://github.com/allenai/allennlp

[3]https://stanfordnlp.github.io/stanfordnlp/

Consolidation

At last, we link all the extracted tuples together to build an object functionality knowledge graph. After consolidation, we removed high frequency noisy objects such as *people, it, one, tool, this* etc. and actions such as *do, try, be, have* etc. Specifically, we remove head of entity type as person which is likely to be stated as subject in many sentences. E.g. "people cut cake" rather than "knife cut cake". Table 5 lists the statistics of extraction results. Furthermore, we adopt PMI to calcualte a prior score for each triplet in order to rerank the frequently or rarely used tool, such as *knife* v.s. *spoon* to cut cake.

$$PMI(h,t) = log\frac{p(h,t)}{p(h)p(t)} \quad (1)$$

Table 5: Extraction results

	#Head Object	#Pair	#Tuple
ConceptNet	3,243	13,857	27,043
Pattern 1	2,921	23,923	84,368
Pattern 2	2,384	15,192	20,068
Pattern 3	4,732	215,819	466,587
Model-based	6,028	60,713	101,257
Consolidation	7,602	278,679	649,060

Evaluation

We evaluate the performance of each method using the Head-Action dataset described in section 3. We employ Pre (precision), Rec (recall) and F1 as metrics to evaluate each method. From the results shown in Table 6, we can see (1) ConceptNet, collected in a crowdsourcing way, shows the best precision but lowest recall. (2) pattern-based method can enlarge the coverage a lot with the help of the large scale Google N-gram dataset. (3) model-based method get better precision than pattern-based method. (4) Consolidation method significantly improves the coverage and achieves the best F1 result.

Table 6: Performance of each extraction method

	Pre	Rec	F1
ConceptNet	0.862	0.152	0.258
Pattern-based	0.632	0.536	0.580
Model-based	0.694	0.255	0.373
Consolidation	0.597	0.624	0.610

5 Object-functionality Prediction

In this section, we study the second research problem: whether the *object* can perform the *actions* on another *object*? We employ the object-functionality knowledge extracted in previous section and predict the plausibility score for a new tuple. The key assumptions are: on one hand, functionality can be identified by visual appearance to some extent, and on the other hand, prior *UsedFor* knowledge is powerful for reasoning the functionality. Motivated by this, we propose a multi-stage model as shown in Figure 4. First, we predict the object categories and functionality through a pre-trained ResNet (He et al. 2016) to extract image feature and a fully connect layer to do prediction. Both category and functionality classification can do simultaneously or separately. Furthermore, we adopt Dismult and Path Ranking Module to inference the plausibility of a tuple <head, action, tail>. Finally, we combine the score through weighted combination.

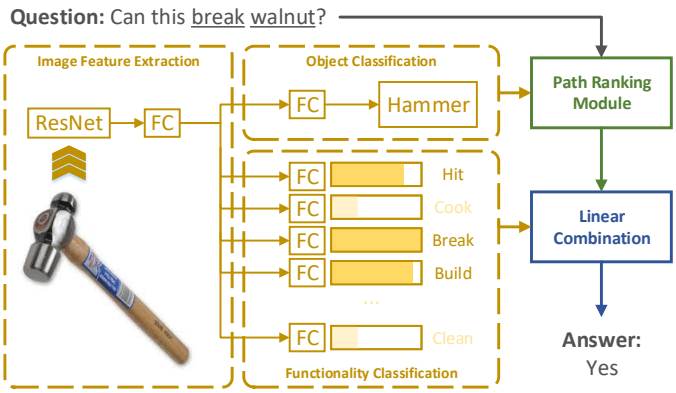

Figure 4: The framework of object-functionality prediction module

Visual appearance module

According to (Azuma, Takiguchi, and Ariki), object appearance can be used to estimate the functionality to a great extent. Therefore, to predict functionality given an object image, we directly use vision features to train models to predict object category as well as the functionality. To be specific, we adopt the image features from a pre-trained convolutional model followed with object and functionality classifiers. In order to acquire the representation of each image, we resize them to $I \in \mathbb{R}^{256\times256}$ first and then use a 5-crop (top-left, top-right, bottom-left, bottom-right and center) on both original and left-right mirror flipped images as the input data $T_I \in \mathbb{R}^{10\times224\times224\times3}$ (the last dimension represents the RGB channels). Next, we feed this augmented input into an ImageNet pre-trained ResNet152(He et al. 2016) model and extract the output of the last convolutional layer $C_I \in \mathbb{R}^{10\times2048}$. After a mean pooling on the first dimension, we finally get the image feature representation: $F_I \in \mathbb{R}^{2048}$.

Then we use a fully connection layer with 512 neurons followed by a batch normalization layer and a ReLU activation layer to map the input image feature into a 512-dimension feature vector. For object classifier, we use a fully connection layer and train as a multi-label classification task. As for functionality classification, we use 20 fully connection layer with binary output for each functionality. These two modules will predict object category as well as a list of recognized functionality.

Dismult inference module

The task is to link prediction given the head object is an image. Dismult(Yang et al. 2015) is an effective knowledge embedding method, which encodes the entities h, t and relation a as representation vector such that valid triplets receive high scores. TransE(Bordes et al. 2013), the translation based embedding methods has a basic principle: $H_p + P \approx T_p$ given the entity and predicate, e.g. E_{Obama}+$P_{president\_of}$≈ $E_{America}$. Our goal is to model commonsense knowledge instead of encyclopedic knowledge. This assumption is hard to satisfy for object-functionality tuples due to one object can perform the same action on multiple various other objects. E.g. *knife* can *cut* many *objects* like cake, banana, paper etc. Different from TransE which only parametrizes the linear relation operators, Dismult adopts bilinear scoring function

$$S(e_h, e_t) \quad = \quad e_h M_a e_t \tag{2}$$

where M_a is the tensor operator and the training objective is to minimize margin-based ranking loss:

$$L = \sum_{positive} \sum_{negative} max\{S_p - S_n + 1, 0\} \tag{3}$$

where S_p is score of positive triplet and S_n is score of negative triplet.

To embed the knowledge graph effectively, we also link consolidated object-functionality knowledge with *IsA* and *Synonym* relation tuples in ConceptNet, which provide intensive linkage between entities to relieve from the sparseness and enrich inference paths.

To train the embedding model, the dimension is set to 200, learning rate as 0.05, the batch size as 128 the negative sample number as 10. We adopted the AdaGrad optimizer and set L2 regularization as 0.0001.

PRA inference module

Besides Dismult, we also employ the path ranking algorithm[4] (Lao, Mitchell, and Cohen 2011) to inference on the consolidated graph described in section 4.4 for the following consideration: PRA(Lao, Mitchell, and Cohen 2011) is an explicit inference and output the possible path as reason. In PRA, given <head, action, tail>, the goal is to predict the plausibility score and paths of the tuple. We train a model for each action by the algorithm 1.

Linear Combination

Finally, we combine the score predicted from visual appearance and inference score using a linear combination:

$$S = \alpha \cdot S_v + \beta \cdot S_i \tag{7}$$

Where S_v is the score from functionality classification model and S_i is the inference score of either PRA or Dismult or both. In our experiment, α is 1.6 and β is 0.57 respectively.

[4]https://github.com/David-Lee-1990/Path-ranking-algorithm

Algorithm 1 Training algorithm

Input: action a, graph tuples {<h, p, t>}
Output: model S_p

1: Iterate all pairs <h, p, t> from the graph, and select the valid tuples for action a with $p = a$ as positive data, and randomly sample tuples from other relations ($p \neq a$) as negative data.
2: For each pair <h, t>, enumerate all possible paths as features $\{F_1, F_2, F_3, \ldots\}$ and $F_i = \{a_1, a_2, a_3\}$. Here we use at most 3-hop relations to learn the possible paths.
3: Then we calculate the probability of each path through the relation

$$Pr(t|h; a) \quad = \quad \frac{a(h,t)}{\sum_i a(h,t_i)} \tag{4}$$

$$Pr(F) \quad = \quad \sum_i Pr(t|h, a_i) \tag{5}$$

where $a(h,t)$ is the count of action a from head h to tail t
4: Next, we calculate the plausibility score of the tuple by a learned weighted linear model

$$S_p = \sum_i w_i \times Pr(F_i) \tag{6}$$

5: Finally, we adopt regularized cross entropy loss to train the model.

Evaluation

In this section, we first evaluate our end-to-end model for object-functionality prediction task and then conduct ablation study on visual appearance module and Inference module separately. For evaluation metric, we use precision, recall and F1 as evaluation metrics.

Prediction task To evaluate the object-functionality prediction, we use 35,860 <head (image), action, tail> tuples as test data and same number as validation from all 678,900 Head(image)-Action-Tail dataset, which share the same split of the image dataset used in the following experiments for visual object and functionality classification. We conduct experiments on 5 different settings: **Functionality Prediction** predicts the validity of tuples base on functionality (action) recognized from the image without taking the tail object into account; **Object Prediction + PRA** classifies the object category through the image and then predict the plausibility of the tuple as <predicted object, action, tail>using PRA; Similarly, **Object Prediction + Dismult** predicts validity using Dismult to inference; Moveover, **Object Classification + Both** predicts score using linear combination of PRA and Dismult to inference; **Our Combination Model** is to combine the scores described in section 5.4.

From Table 7, we get three insights: (1) object classification followed by either PRA or Dismult inference algorithm performs worse than prediction by Functionality classification model directly, which shows it is effective to rec-

Table 7: Result of object-functionality prediction

	Pre	Rec	F1
Functionality Prediction	0.741	0.771	0.756
Object Prediction + PRA	0.652	0.388	0.486
Object Prediction + Dismult	0.671	0.633	0.651
Object Prediction + Both	0.747	0.8139	0.7789
Our Combination Model	0.778	0.845	0.810

Table 8: Result of visual classification models

	Pre	Rec	F1
Functionality classification only model	0.771	0.701	0.734
Functionality + object classification model	0.776	0.691	0.731

ognize the functionality from image appearance with supervised data, while object classification followed by a combined inference performs better than prediction by Functionality classification model; (2) Dismult performs better than PRA especially on Recall, which shows that implicit representation by knowledge embedding encodes more information; (3) the combination with both methods get the best results. According to our analysis, object classification predicts the object category first, which leads to some error and propagate to the next inference step. E.g. "key" is recognized as "cutter". While for other cases, although several objects are mis-classified, e.g. *knife* is recognized as *cutter*, we find that they have similar appearance and functionality. Moreover, 'Functionality Classification' directly learns the functionality through raw image, which can capture various detailed low-level features. The second assumption mentioned in Section 5 also holds true in most cases. "Knife" can "cut" almost everything except something hard (like "diamond").

Ablation study: visual module We study the performance of functionality classification module. We split the dataset into 3 parts with 630 testing images and 630 validation, and the result as training set. We use the 330 positive Head-Action pairs to annotate functionality of each image. We compare the results of predicting functionality and object categories in: (1) Functionality classification only model; (2) Functionality + Object classification joint model. From the results shown in Table 8, we get the comparable result from both methods, and we apply the Functionality classification only model in experiment.

Ablation study: Dismult module We train Dismult model to embed either ConceptNet or consolidated graph separately, and compared the performance using the object functionality prediction task. Table 9 shows that with the large consolidated graph, the prediction result outperforms that of using ConceptNet graph.

Ablation study: PRA module We also evaluate PRA module on ConceptNet and our consolidated graph on two

Table 9: Result of Dismult

Graph	Pre	Rec	F1
ConceptNet	0.564	0.516	0.539
Consolidated	0.671	0.633	0.651

Table 10: Result of PRA

Data source	Graph	Pre	Rec	F1
Random sample	ConceptNet	0.793	0.073	0.134
Head-Action-Tail	ConceptNet	0.248	0.036	0.063
Random sample	Consolidated	0.912	0.396	0.552
Head-Action-Tail	Consolidated	0.729	0.41	0.525

test dataset: randomly sampled data from each graph and 3,900 Head-Action-Tail tuples. From the results in Table 10, we can see that by using our consolidated graph, both the coverage(Recall) and precision outperforms compared the result using ConceptNet graph, which verifies the effectiveness of our extraction method. Simultaneously, the Head-Action-Tail dataset has different distribution and thus has worse performs than the randomly sampled dataset from the original graph. Besides, we analysis the performance of each action in details. Action *cut* performs the best, while *roll repair*, *decorate* performs the worst due to knowledge graph sparseness. Given the action "cut" has rich and dense tuples, the performance has achieved the best. Along with more knowledge, it is easier for inference.

Case study We analyse several cases and find two insights: 1) If the object classification is incorrect, the functionality prediction may be correct, e.g. "mallet" is classified as "hammer", and "cutter" is predicted as "knife". Despite inaccurate object classification, we still get correct prediction due to the same functionality of these objects with similar appearance. 2) Even though the object classification is correct, the triplet may not. As discussed in Ablation study, although the function "cut" prediction gets high accuracy, there are still some errors. One typical error example is that "scissors cut cake": although "scissors" can "cut" many things, people rarely use it to "cut cake".

6 Conclusion

We propose two research problems: (1) What *functionality/actions* an *object* has? (2) Whether the *object* can perform *action* on another object? Object-functionality knowledge is important in robotic community for robots to complete a task. We first mine object functionality knowledge from text data, and then introduce a new task for object-functionality prediction to simulate what the robot perceive. According to our experiment, we found 1) the extraction method can increase the graph in a large scale, and the extracted tuples are effective for further link prediction task and functionality(image as head entity) prediction task. 2) Dismult models with implicit representation perform better than explicit PRA model for inference.

References

Azuma, R.; Takiguchi, T.; and Ariki, Y. Estimation of object functions using visual attention.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, 2787–2795.

Chao, Y.-W.; Wang, Z.; Mihalcea, R.; and Deng, J. 2015. Mining semantic affordances of visual object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4259–4267.

Chen, X.; Chen, M.; Shi, W.; Sun, Y.; and Zaniolo, C. 2019. Embedding uncertain knowledge graphs. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*.

Chen, X.; Shrivastava, A.; and Gupta, A. 2013. Neil: Extracting visual knowledge from web data. In *Proceedings of the IEEE International Conference on Computer Vision*, 1409–1416.

Gibson, J. J. 2014. *The ecological approach to visual perception: classic edition*. Psychology Press.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Krishna, R.; Zhu, Y.; Groth, O.; Johnson, J.; Hata, K.; Kravitz, J.; Chen, S.; Kalantidis, Y.; Li, L.-J.; Shamma, D. A.; et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision* 123(1):32–73.

Lao, N.; Mitchell, T.; and Cohen, W. W. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 529–539. Association for Computational Linguistics.

Lin, X., and Parikh, D. 2015. Don't just listen, use your imagination: Leveraging visual common sense for non-visual tasks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2984–2993.

Lin, Y.; Michel, J.-B.; Aiden, E. L.; Orwant, J.; Brockman, W.; and Petrov, S. 2012. Syntactic annotations for the google books ngram corpus. In *Proceedings of the ACL 2012 system demonstrations*, 169–174. Association for Computational Linguistics.

Lu, P.; Ji, L.; Zhang, W.; Duan, N.; Zhou, M.; and Wang, J. 2018. R-vqa: learning visual relation facts with semantic attention for visual question answering. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1880–1889. ACM.

Mitchell, T.; Cohen, W.; Hruschka, E.; Talukdar, P.; Yang, B.; Betteridge, J.; Carlson, A.; Dalvi, B.; Gardner, M.; Kisiel, B.; et al. 2018. Never-ending learning. *Communications of the ACM* 61(5):103–115.

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

Petrochuk, M., and Zettlemoyer, L. 2018. Simplequestions nearly solved: A new upperbound and baseline approach. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 554–558.

Qi, P.; Dozat, T.; Zhang, Y.; and Manning, C. D. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, 160–170. Brussels, Belgium: Association for Computational Linguistics.

Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A.; and Pineau, J. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 3776–3783. AAAI Press.

Speer, R., and Havasi, C. 2012. Representing general relational knowledge in conceptnet 5. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*.

Stanovsky, G.; Michael, J.; Zettlemoyer, L.; and Dagan, I. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 885–895.

Tandon, N.; De Melo, G.; Suchanek, F.; and Weikum, G. 2014. Webchild: Harvesting and organizing commonsense knowledge from the web. In *Proceedings of the 7th ACM international conference on Web search and data mining*, 523–532. ACM.

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

Yang, B.; Yih, S. W.-t.; He, X.; Gao, J.; and Deng, L. 2015. Embedding entities and relations for learning and inference in knowledge bases.

Yao, B.; Ma, J.; and Fei-Fei, L. 2013. Discovering object functionality. In *Proceedings of the IEEE International Conference on Computer Vision*, 2512–2519.

Yatskar, M.; Ordonez, V.; and Farhadi, A. 2016. Stating the obvious: Extracting visual common sense knowledge. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 193–198.

Young, T.; Cambria, E.; Chaturvedi, I.; Zhou, H.; Biswas, S.; and Huang, M. 2018. Augmenting end-to-end dialogue systems with commonsense knowledge. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Zhu, Y.; Fathi, A.; and Fei-Fei, L. 2014. Reasoning about object affordances in a knowledge base representation. In *European Conference on Computer Vision*, 408–424. Springer.

Zhu, Y.; Zhao, Y.; and Chun Zhu, S. 2015. Understanding tools: Task-oriented object modeling, learning and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2855–2864.