IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2023, London, UK. Cite as RA-L paper.

Recommending Fine-grained Tool Consistent with Common Sense Knowledge for Robot

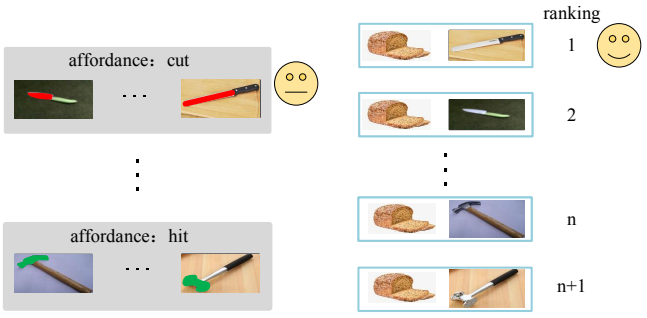
Jianjia Xin[1] Lichun Wang[2] Shaofan Wang[3] Yukun Liu[4] Chao Yang[5] and Baocai Yin[6]

***Abstract*—When robots carry out task, selecting an appropriate tool is necessary. The current research ignores the fine-grained characteristic of tasks, and mainly focuses on whether the task can be completed. Little consideration is paid for the object being manipulated, which affects the task completion quality. In order to support task oriented fine-grained tool recommendation, based on common sense knowledge, this paper proposes Fine-grained Tool-Task Graph (FTTG) to describe multi-granularity semantics of tasks, tools, objects being manipulated and relationships among them. According to FTTG, a Fine-grained Tool-Task (FTT) dataset is constructed by labeling images of tools and objects being manipulated with the defined semantics. A baseline method named Fine-grained Tool Recommendation Network (FTR-Net) is also proposed in this paper. FTR-Net gives coarse-grained and fine-grained semantic predictions by simultaneously learning the common and special features of the tools and objects being manipulated. At the same time, FTR-Net constrains the distance between features of the well matched tool and object more smaller than that of those unmatched. The constraint and the special feature ensure FTR-Net provide fine-grained tool recommendation. The constraint and the common feature ensure FTR-Net provide coarse-grained tool recommendation when the fine-grained tools are not available. Experiments show that FTR-Net can recommend tools consistent with common sense whether on test data sets or in real situations.**

***Index Terms*—Computer Vision for Automation, Data Sets for Robotic Vision, Deep Learning for Visual Perception.**

I. Introduction

In recent years, artificial intelligence algorithms have flourished. As an essential carrier of intelligent algorithms, robots attract much attention. Robots can replace human labor and serve human beings by completing specified tasks. When a robot automatically executes a task, it needs to select appropriate tools, and the selection results will affect the quality of task completion. However, the current research mainly considers which tool can be used to complete the task, and rarely considers whether the selected tool is suitable for manipulating the object in terms of task completion quality. Some existing methods select tools by analyzing their affordances. In Fig. 1 (a), the affordances of "fruit knife" and "bread knife" are both "cut", so the two kinds of knives could be selected as candidate tools for the same task. In fact, according to common sense, "bread knife" is a more appropriate choice for the "cutting bread" task. The reason is that the "bread knife" will not flatten the bread when cutting fluffy bread because of its serrated blade, while the "fruit knife" will flatten the bread when cutting bread. If ignoring the properties of the manipulated objects and only considering the availability of tools, tools that are not suitable for performing tasks will be recommended.



(a) A set of candidate tools having the same affordance

(b) One candidate tool being the most suitable

Fig. 1: Selecting candidate tool for completing the "cutting bread" task. Bread is the object to be manipulated. The "fruit knife", "bread knife" and other tools which own affordance "cut" are selected as candidate tools based on predicting the affordances of tools. The first ranked "bread knife" is selected as the only candidate tool based on evaluating the relation between the tools and the bread.

In this paper, the tool and the object being manipulated are considered together to evaluate the rationality of their interaction. In Fig. 1 (b), according to the feature distances between bread and tools, the ranking of tools is given for the "cutting bread" task and the first ranked "bread knife" is selected as the only candidate tool. The smaller the feature distance, the higher the ranking. When considering the objects to be manipulated, different tools should be selected for tasks such as "cutting bread" and "cutting meat" due to the different attributes of the objects to be operated on. This paper defines the tasks related with same operation but different objects as fine-grained task. When the manipulated objects are ignored, fine-grained tasks such as "cutting bread" and "cutting meat" can be regarded as the same task, namely "cutting task", which is defined as coarse-grained task in this paper. According to the common sense knowledge of human using tools, we define the optimal tool for accomplishing the fine-grained task as fine-grained tool. A coarse-grained task corresponds to a set of fine-grained tasks which have similar requirements for the affordance of tools. When the fine-grained tool corresponding to a fine-grained task is unavailable, the tools corresponding to the other fine-grained tasks belonging to the same coarse-grained task can be used as the coarse-grained tool for the

The authors are with the Beijing Key Laboratory of Multimedia and Intelligent Software Technology, Beijing Artificial Intelligence Institute, Faculty of Information Technology, Beijing University of Technology, Beijing. 100124, China (e-mail: xinjianjia@emails.bjut.edu.cn; wanglc@bjut.edu.cn; wangshaofan@bjut.edu.cn; liuyukun@emails.bjut.edu.cn; yangchaoyc@emails.bjut.edu.cn; ybc@bjut.edu.cn).*Corresponding author: Lichun Wang.*

fine-grained task. As shown in Fig. 1 (b), the second ranked "fruit knife" is a coarse-grained tool for the "cutting bread" task.

A coarse-grained task corresponds to multiple kinds of tools but a fine-grained task corresponds to few types of tools. We propose Fine-grained Tool-Task Graph (FTTG) to describe multi-granularity semantics of tasks, tools, objects being manipulated and relationships among them. According to FTTG, we collect images of tools and objects being manipulated from the internet to construct Fine-grained Tool-Task (FTT) dataset. Simultaneously, this paper proposes Fine-grained Tool Recommendation Network (FTR-Net) to recommend appropriate tools for interaction tasks by computing the distance between embedding features of the tool and the manipulated object images.

In order to verify the effectiveness of the recommended tools for fine-grained tasks, this paper establishes three test cases corresponding to different types of application scenarios. The first case requires recommending tool for a definite fine-grained task from a set of fine-grained tools which are corresponded to different fine-grained tasks. The second case requires recommending tool for a definite fine-grained task from a set of fine-grained tools which are corresponded to the same coarse-grained task. The third case requires recommending tool for a definite fine-grained task from a set of tools including a coarse-grained tool and multiple tools corresponded to the other coarse-grained tasks. In the above three test cases, FTR-Net has achieved good results.

In summary, the main contributions of this paper include the following :

- Propose fine-grained tool recommendation which recommends the most appropriate tool to complete a certain task with higher quality by considering the fine-grained nature of the task.
- Propose FTTG which describes multi-granularity semantics of tasks, tools, objects being manipulated and relationships among them based on common sense knowledge, and provide dataset FTT for verifying the fine-grained tool recommendation.
- Propose a baseline for the fine-grained tool recommendation, namely FTR-Net. It learns embedded features of tools and objects being manipulated, and recommends the most appropriate tool for a definite task by calculating the distances between the embedded features.

II. Related Work

Choosing Tools in Robotics. Choosing tools based on affordance is a kind of common method. Affordance refers to the tool's actionable properties [1], which usually indicates the tool's function [2]. Some researches regard affordance as a pixel-wise classification problem, which predicts an affordance label for each pixel in the tool part. For example, the blade of a knife can be used to perform a "cut" action, which indicates that the knife has the function of cutting. Myers et al. [3] used hand-designed features to predict the tool's affordance. Nguyen et al. [4] used encoder-decoder architecture to learn features from RGB-D data automatically which achieved better results than hand-designed features. Chaudhary et al. [5], Do et al. [6], Zhao et al.[7] and Chen et al. [8] used the popular networks such as FCN [9], Mask R-CNN [10] to build segmentation networks to predict the affordance semantics. The above methods require pixel-wise affordance labels during the training phase. However, the workload of pixel-wise labeling is heavy, some methods [11], [12] used weakly supervised learning to predict pixel-wise affordance semantic. Using the methods based on labeling affordance, fine-grained tools can not be clearly identified for fine-grained tasks. For instance, the blades of "fruit knife" and "bread knife" are both predicted "cut", and can not be further distinguished. For the fine-grained task "cutting bread", the above two kinds of knives can not be prioritized well.

Except choosing tools based on affordance, some researches focus on the inherent characteristics of the tool (such as material, volume) or the physical information displayed when the tool is used to perform tasks (such as force, speed) [13], [14]. Besides, some researches compare the similarity between tools known appropriate for a certain task and tools in unknown environments. In the training phase, the robot learns the characteristics of the tools needed to complete the task. In the testing phase, the robot calculates the similarity of learned characteristics between seen tools and unseen tools [15], [16], [17], [18], [19], [20]. For example, Abelha et al. [17] computed fitting degree between tools based on 3D point cloud and quantized the fitting degree.

Using Common Sense Knowledge in Robotics. Common sense knowledge plays an important role in robotics. Kunze et al. [21] applied common sense knowledge to household robots. Kaiser et al. [22] and Al-Moadhen et al. [23] used common sense knowledge for robot planning. Common sense knowledge is also widely used in robot navigation [24], affordance modeling [25], [26], manipulation [27]. In this paper, the relationship between fine-grained tasks, tools and manipulated objects conforms to the common sense knowledge of human daily life.

III. Problem Statement

Given a fine-grained task T, an object image I_o and a set of tool images $\{I_t\}$, the aim is to recommend the fine-grained tool image $\hat{I}_t$ as a candidate used for operating on the object. Learning to recommend the fine-grained tool means finding a function $f : (\{I_t\}, I_o) \longmapsto \hat{I}_t$.

IV. Fine-grained Tool-Task Dataset

In order to address the problem of recommending tools for fine-grained tasks, we propose FTTG to describe multi-granularity semantics of tasks, tools, objects being manipulated and the relationships among them. In FTTG, the semantics are described on two levels, coarse-grained parent class and fine-grained subclass. The statics of semantic concepts are shown in Table I. The semantic concepts and relationships are organized by the directed graph shown in Fig. 2, represented by nodes and edges respectively. Each directed edge in Fig. 2 describes the relationship between a pair of semantic concepts. The relationships in FTTG include "cut", "catch the liquid

TABLE I: Statistics of tasks, tools and objects being manipulated with different levels of semantics in FTTG.

Parent task	Subclass task	Parent tool	Subclass tool	Parent class object being manipulated	Subclass object being manipulated
cutting task	cutting cucumber	cutting tool	slicing knife	object being cut	cucumber
	cutting banana		fruit knife		banana
	cutting bread		bread knife		bread
	cutting cake				cake
	cutting pizza		shake knife		pizza
			hob knife		
containing task	containing tea	containing tool	teacup	object for holding beverages	teapot
	containing red wine		goblet		wine bottle
	containing coffee		coffee cup		coffee maker
cropping task	cropping paper	cropping tool	small scissor	object being cropped	paper
	cropping tape				tape
	cropping cotton material		tailor scissor		cotton material
	cropping rope				rope
hitting task	hitting iron nail	hitting tool	claw hammer	object being hit	iron nail
	hitting raw meat		meat hammer		raw meat
storing task	storing cloth	storing tool	wardrobe	object being stored	cloth
	storing book		bookshelf		book

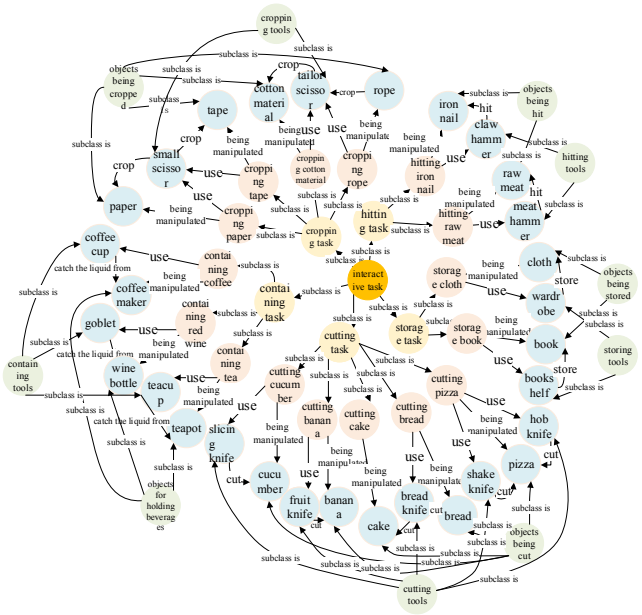
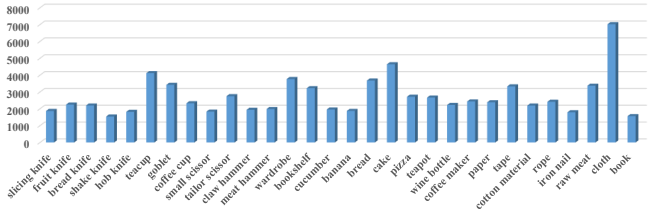
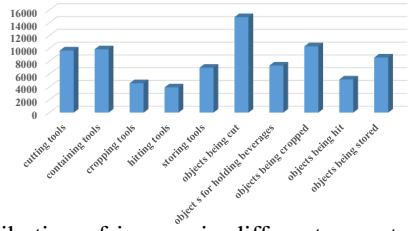


Fig. 2: Visualization of FTTG.



(a) Distribution of images in different subclasses



(b) Distribution of images in different parent classes

Fig. 3: Statistics of the labeled images.

from", "crop", "hit", "store" which connect the subclass tool and the subclass object being manipulated; "subclass is" which connects the parent class task and the subclass task, the parent class tool and the subclass tool, the parent class object being manipulated and the subclass object being manipulated; "use" which connects the subclass task and the subclass tool; "being manipulated" which connects the subclass task and the subclass object being manipulated.

According to FTTG, images of the subclass tool and the subclass object being manipulated are collected from the internet. The collected images are manually labeled with multi-granularity semantic labels. Statistics of subclass and parent class images are shown in Fig. 3, the number of images in each subclass is not less than 1500, and the number of images in all subclasses is 81358.

In order to learn the knowledge related to tool recommendation, we construct FTT dataset based on FTTG, which includes paired images in the form of $< tool\ image, manipulated\ object\ image >$. In FTT dataset, there are three kinds of samples, positive, neutral and negative sample. A sample is considered as positive if the subclasses to which the two images belong are connected by a directed edge in FTTG, otherwise, the sample is considered as neutral or negative. If the subclasses to which the two images belong have common grandparent node in FTTG, the sample composed of the two images is regarded as a neutral sample, otherwise it is regarded as a negative sample.

Fig. 4 shows examples of the three types of samples. Fig. 4 (a) shows a positive sample, "bread knife" and "bread" is connected with a directed edge "cut" in FTTG, so the sample is a positive sample. Fig. 4 (b) shows a neutral sample, on one hand there is no directed edge between "fruit knife" and "bread" in FTTG, on the other hand, "fruit knife" and "bread" have common grandparent node "cutting task" in FTTG. Fig. 4 (c) shows a negative sample, "teacup" and "bread" have no

directed edge between them and have no common grandparent node in FTTG.

(a) Positive sample (b) Neutral sample (c) Negative sample

Fig. 4: Sample examples for "cutting bread" task.

According to FTTG, the two images belonging to a positive sample have same parent node which is a subclass task, so the tool is the fine-grained tool to complete the fine-grained task. The two images belong to a neutral sample have same grandparent node which is a parent class task, indicating their associated fine-grained task belongs to the same coarse-grained task, so the tool is a coarse-grained tool to complete the fine-grained task. For the tool and the object included in a negative sample, the tool is irrelevant with the fine-grained task and coarse-grained task corresponding to the object.

According to the above way of organizing samples, the proposed FTT dataset includes 1639500 samples, consisting of 374000 positive, 1156500 negative and 109000 neutral samples. FTT dataset is extensible and convenient to append new tasks, tools and objects being manipulated because the data are organized according to the directed graph FTTG. When expanding FTT, graph nodes and directed edges are first added to FTTG to provide semantic definition, then corresponding paired images can be added to FTT dataset. The code for automatically constructing samples is available online[1]. The FTT dataset is available online[2].

V. Fine-grained Tool Recommendation Network

FTR-Net learns semantics described in FTTG to infer tools suitable for fine-grained tasks. As shown in Fig. 5, FTR-Net first extracts features of the tool image and the image of the manipulated object, and then predicts their coarse-grained semantics, fine-grained semantics and the relationship between them.

FTR-Net uses ResNet-50 [28] to extract features of images in positive and negative samples. For each sample, the subclass tool image and the image of the manipulated object are correspond to head and tail element respectively. With ResNet-50, visual features $\varphi_{\mathrm{h}}^{(i)}$ and $\varphi_{\mathrm{t}}^{(i)}$ are computed for head and tail element of the i-th positive sample in the batch. Features $\gamma_{\mathrm{h}}^{(j)}$ and $\gamma_{\mathrm{t}}^{(j)}$ are computed for head and tail element of the j-th negative sample in the batch.

Following, the hierarchical categorization for the image is explained by taking the i-th positive sample's head as an example. The $\varphi_{\mathrm{h}}^{(i)}$ is input to the fully connected layers. Then, the sigmoid activation function $g(\cdot)$ is used to obtain the multi-granularity semantic prediction of the image:

$$p_{\mathrm{h}}^{(i)} = g\left(W_g \varphi_{\mathrm{h}}^{(i)}\right) \tag{1}$$

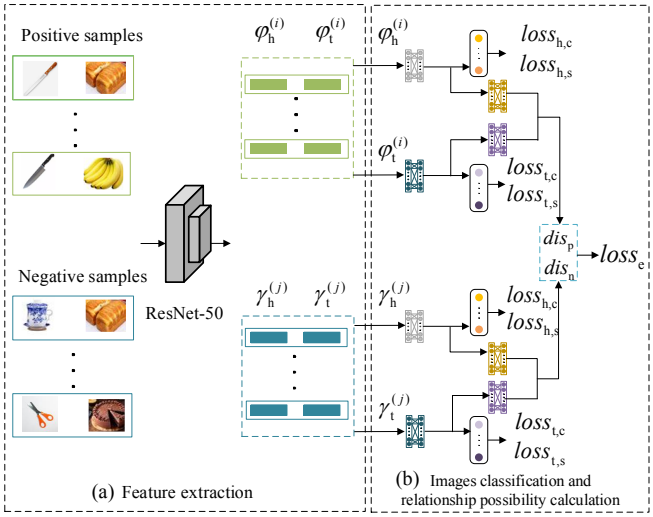


Fig. 5: The overall framework of FTR-Net.

W_g represents the parameters in the fully connected layers. Fig. 6 shows the structure of the hierarchical prediction $p_{\mathrm{h}}^{(i)}$. The first C components of $p_{\mathrm{h}}^{(i)}$ give the probability of the image belonging to each parent category, and each element is denoted as $y_{\mathrm{h,c}}^{(d)}$. The last S components of $p_{\mathrm{h}}^{(i)}$ give the probability of the image belonging to each subclass category, and each element is denoted as $y_{\mathrm{h,s}}^{(k)}$. The binary cross-entropy loss for the hierarchical categorization is calculated by

$$loss_{\mathrm{h,c}}^{(i)} = -\frac{1}{C}\sum_{d=1}^{C}\left[\hat{y}_{\mathrm{h,c}}^{(d)} \log y_{\mathrm{h,c}}^{(d)} + \left(1-\hat{y}_{\mathrm{h,c}}^{(d)}\right)\log\left(1-y_{\mathrm{h,c}}^{(d)}\right)\right] \tag{2}$$

$$loss_{\mathrm{h,s}}^{(i)} = -\frac{1}{S}\sum_{k=1}^{S}\left[\hat{y}_{\mathrm{h,s}}^{(k)} \log y_{\mathrm{h,s}}^{(k)} + \left(1-\hat{y}_{\mathrm{h,s}}^{(k)}\right)\log\left(1-y_{\mathrm{h,s}}^{(k)}\right)\right] \tag{3}$$

$\hat{y}_{\mathrm{h,c}}^{(d)}$ and $\hat{y}_{\mathrm{h,s}}^{(k)}$ represent the value of the d-th dimension in the parent class ground truth vector and the k-th dimension in the subclass ground truth vector respectively. The hierarchical visual categorization loss of the head element is the sum of $loss_{\mathrm{h,c}}^{(i)}$ and $loss_{\mathrm{h,s}}^{(i)}$:

$$loss_{\mathrm{h}}^{(i)} = loss_{\mathrm{h,c}}^{(i)} + \lambda_{\mathrm{s}} loss_{\mathrm{h,s}}^{(i)} \tag{4}$$

λ_{s} is the weight factor. In the same way, $loss_{\mathrm{h}}^{(j)}$ for the j-th negative sample can be calculated. We average the classification loss of all head elements in the batch and record it as $loss_{\mathrm{h}}$. Similarly, $loss_{\mathrm{t}}$ represents the average visual categorization loss of tail elements.

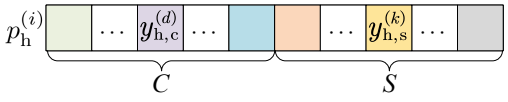


Fig. 6: Dimension and elements of vector $p_{\mathrm{h}}^{(i)}$.

[1]https://github.com/AXINLETTER/FTTG.git

[2]https://drive.google.com/drive/folders/1uvikHzAzd5XCtUXIGZRUWNFqwDJTvquY?usp=sharing

Using the learned features, the distance of images' embedding features can be computed. For the i-th positive sample, distance of images' embedding features is denoted as $dis_{\mathrm{p}}^{(i)}$, which is calculated by

$$dis_{\mathrm{p}}^{(i)} = ||W_e\varphi_{\mathrm{h}}^{(i)} - W_p\varphi_{\mathrm{t}}^{(i)}||^2 \quad (5)$$

For the j-th negative sample, distance of images' embedding features is denoted as $dis_{\mathrm{n}}^{(j)}$, which is calculated by

$$dis_{\mathrm{n}}^{(j)} = ||W_e\gamma_{\mathrm{h}}^{(j)} - W_p\gamma_{\mathrm{t}}^{(j)}||^2 \quad (6)$$

W_e and W_p represent the parameters in the embedding layers.

In order to learn correct relationship between the object being manipulated and it's corresponding fine-grained tool, the average embedding distance of positive samples is constrained to be much smaller than that of negative samples, which is realized with

$$loss_{\mathrm{e}} = \max(0, \frac{1}{M}\sum_{i=1}^{M} dis_{\mathrm{p}}^{(i)} - \frac{1}{N}\sum_{j=1}^{N} dis_{\mathrm{n}}^{(j)} + margin) \quad (7)$$

$margin$ is a hyperparameter. M and N represent the number of positive and negative samples in the batch.

The total loss is linear combination of $loss_{\mathrm{h}}$, $loss_{\mathrm{t}}$ and $loss_{\mathrm{e}}$:

$$loss_{\mathrm{all}} = \lambda_{\mathrm{h}}loss_{\mathrm{h}} + \lambda_{\mathrm{t}}loss_{\mathrm{t}} + \lambda_{\mathrm{e}}loss_{\mathrm{e}} \quad (8)$$

λ_{h}, λ_{t} and λ_{e} are weight factors to be specified manually. The weights of the neural network are optimized during training.

VI. Experiment

A. Training and Test Sets for Tool Recommendation

In order to verify the performance of tool recommendation for the 16 fine-grained tasks in Table I, FTT dataset is divided into training and test sets. The training set includes 340000 positive samples and 640500 negative samples. The test set includes 34000 positive, 516000 negative, and 109000 neutral samples. The set of images in test samples has no intersection with the set of images in training samples.

The test set is divided into three subsets (test set A, B and C) which is corresponding to three test cases and each of them is corresponding with 16 fine-grained tasks. Test set A and B are used to evaluate the fine-grained tool recommendation for fine-grained tasks, test set C is used to evaluate the coarse-grained tool recommendation for fine-grained tasks. For the recommendation of a certain fine-grained tool, we set up test units and each of them contains several test samples sharing the same manipulated object image. Test set A contains 17000 units, each containing one positive, multiple negative and neutral samples. The test set A contains 17000 positive, 38000 neutral, and 183000 negative samples. Test set B contains 17000 units, each containing one positive and multiple neutral samples. The test set B contains 17000 positive and 38000 neutral samples. Test set C contains 33000 units, each containing one neutral and multiple negative samples. The test set C contains 33000 neutral and 333000 negative samples. Fig. 7 shows three test units corresponding to the "cutting bread" task in different test sets.

Fig. 7 (a) gives an example in test set A. The fine-grained tool "bread knife" marked by a red border forms a positive sample with "bread". The tools "slicing knife", "fruit knife", "shake knife" and "hob knife" which marked by green borders can be used to complete same coarse-grained "cutting task" and form neutral samples with "bread". The other tools used to complete other coarse-grained tasks such as "teacup" and "scissors" marked by yellow borders form negative samples with "bread".

Fig. 7 (b) gives an example in test set B. The fine-grained tool "bread knife" forms a positive sample with "bread". The tools "fruit knife", "shake knife" and "hob knife" which can be used to complete same coarse-grained "cutting task" form neutral samples with "bread".

Fig. 7 (c) gives an example in test set C. The coarse-grained tool "fruit knife" forms a neutral sample with "bread". The other tools used to complete other coarse-grained tasks such as "teacup" and "scissors" etc. form negative samples with "bread".

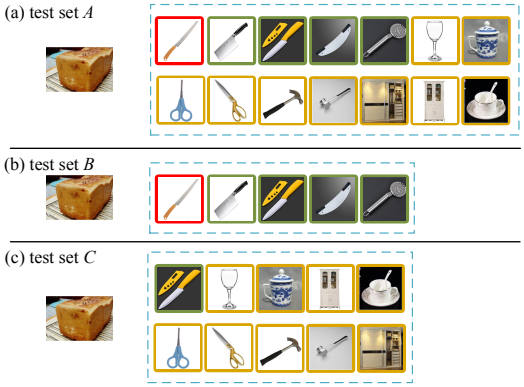


Fig. 7: Units in test sets for "cutting bread" task.

B. Test Process and Performance of FTR-Net

For a certain fine-grained task, a corresponding unit consists of multiple samples. Each sample is composed of a tool image and an image of manipulated object. During test, the distance between embedding features of the tool image and the manipulated object image is calculated for each sample. The tool corresponding to the smallest distance in the unit is recommended as the candidate tool for the task.

We use recommendation accuracy (Acc_r) to evaluate model performance. Acc_r represents the proportion of the fine-grained tool that rank first in all prediction for the test units corresponding to a certain fine-grained tool recommendation task and is computed as

$$Acc_r = \frac{A}{U} \times 100\% \quad (9)$$

where A represents the number of fine-grained tools that rank first, U is the number of units corresponding to a certain fine-grained tool recommendation.

The experiment results on the three test sets are shown in Table II. The upper, middle and lower columns in Table II show the results on test set A, B, and C, respectively. FTR-Net

TABLE II: The tool recommendation accuracy (%) on test set A, B, C are separated by double horizontal lines from top to bottom. FTR-Net (Euc) indicates that the distance of images' embedding features is Euclidean distance. FTR-Net (Cos) indicates that the distance of images' embedding features is cosine distance.

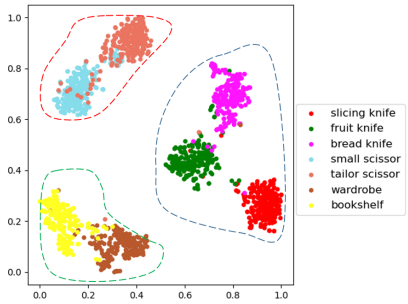
Task	cutting cucumber	cutting banana	cutting bread	cutting cake	cutting pizza	containing tea	containing red wine	containing coffee	cropping paper	cropping tape	cropping cotton material	cropping rope	hitting iron nail	hitting raw meat	storing cloth	storing book	Avg.
FTR-Net (Euc)	90.3	43.0	67.2	66.4	96.0	54.3	87.2	75.3	44.6	46.6	65.7	65.6	75.8	78.9	97.2	91.4	71.6
FTR-Net (Cos)	86.2	56.5	78.4	78.9	83.7	27.5	83.7	72.6	70.8	85.1	72.2	79.8	94.3	93.3	93.1	91.5	78.0
FTR-Net (Euc)	90.2	39.3	67.2	69.4	95.4	55.0	90.1	78.5	47.7	48.6	68.6	67.4	91.9	97.2	97.9	95.3	75.0
FTR-Net (Cos)	87.4	55.1	77.8	77.7	85.0	29.6	84.7	75.9	77.9	88.6	71.6	81.0	98.1	95.5	94.0	93.6	79.6
FTR-Net (Euc)	51.1	49.0	50.7	50.7	97.7	97.6	94.1	99.6	83.8	94.6	93.6	88.1	11.9	13.3	100.0	58.5	70.9
FTR-Net (Cos)	99.6	99.0	97.3	97.6	99.1	97.7	97.6	99.2	93.5	97.7	96.4	92.3	89.6	93.9	99.7	94.8	96.6

is recorded as FTR-Net (Cos) while the distance of images' embedding features is cosine distance. The cosine distances of embedding features $dis_{\mathrm{p}}^{(i)}$ and $dis_{\mathrm{n}}^{(j)}$ are calculated by

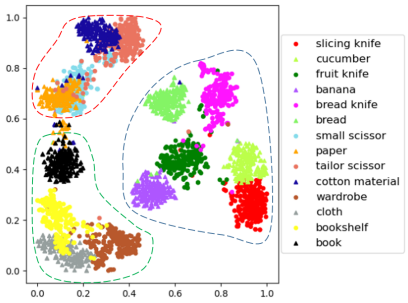
$$dis_{\mathrm{p}}^{(i)} = 1 - cos(W_e\varphi_{\mathrm{h}}^{(i)}, W_p\varphi_{\mathrm{t}}^{(i)}) \tag{10}$$

$$dis_{\mathrm{n}}^{(j)} = 1 - cos(W_e\gamma_{\mathrm{h}}^{(j)}, W_p\gamma_{\mathrm{t}}^{(j)}) \tag{11}$$

Compared with using Euclidean distance of images' embedding features, the performance on most fine-grained tasks is improved in the case of using cosine distance, and the average recommendation accuracy is improved with 6.4%, 4.6% and 25.7% on the three test sets respectively. FTR-Net (Cos) performs better than FTR-Net (Euc) especially on the test set C, because cosine distance shows more significantly difference compared with Euclidean distance while evaluating the similarity of images belonging to different categories. The experiment results show that FTR-Net can recommend fine-grained tools for fine-grained tasks, and can effectively retrieve coarse-grained tools when the fine-grained tool is not available.



(a) Tools



(b) Tools and manipulated objects

Fig. 8: T-SNE of embedding features of tools and manipulated objects. Colorful dots represent tool features, and colorful triangles represent manipulated object features.

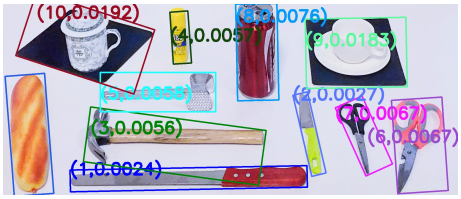
In order to analyze whether the visual features extracted by FTR-Net support tool recommendations for tasks with different granularity, we choose images of the tool and object being manipulated corresponding to following fine-grained tasks, "cutting cucumber", "cutting banana", "cutting bread", "cropping paper", "cropping cotton material", "storing cloth" and "storing book". Using T-SNE to visualize the high-dimensional visual features of tool images and the visualization is shown in Fig. 8 (a). Colorful dots represent tool features and each color is corresponding to a kind of fine-grained tool. The dots with same color are clustered well and dots with different colors are well detached, which indicates that FTR-Net can distinguish tool at a fine-grained level. In Fig. 8 (a), the three regions marked with dotted lines includes tools corresponding to the same coarse-grained task. The tools within the region delineated by the dotted line are closer to each other than the tools outside the region, which indicates that FTR-Net can extract common features for tools corresponding to the same coarse-grained task.

Fig. 8 (b) updated the T-SNE visualization on the basis of Fig. 8 (a) by adding the high-dimensional visual features of manipulated object images that are represented with colorful triangles. The tools are close to the objects corresponding to a certain fine-grained task, which indicates the tool and the object can be matched well in the case of appointing a fine-grained task. The tools and the objects corresponding to same coarse-grained task locate within the region marked by dotted lines, which indicates the tool and the object are relative close. So FTR-Net can recommend an appropriate candidate tool for a certain coarse-grained task.

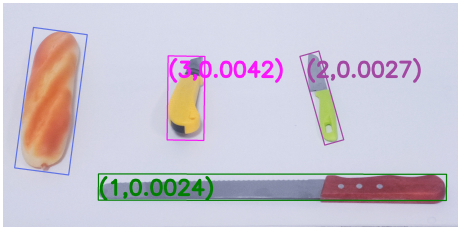
C. Interactive Tool Recommendation in Actual Situations

In order to to test the performance of recommending tools in actual situations, we take the "cutting bread" task as an example. We set up three cases in actual environment that comply to the same principle of the test set in FTT. In order to test the generalization of FTR-Net, some tools unavailable in FTT dataset, including "glue stick", "coke can" and "box cutter" , are placed in the actual scenario.

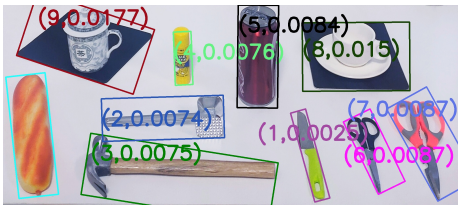
Fig. 9 shows experiments in the three cases. The bounding boxes in the image are generated according to an edge detection algorithm. Using the minimum and maximum pixel coordinates of the bounding box, a box parallel to horizontal and vertical directions is calculated to crop the image region including the bread or tool from the captured image. The bread image and each tool image form a sample. All samples are



(a) test case A



(b) test case B



(c) test case C

Fig. 9: Ranking of recommendation and distance between embedding features (lower is better) for the "cutting bread" task in actual situations.

input to FTR-Net. Calculating distance between the embedding features of the tools and the "bread", then the tools are ranked in ascending order according to the distances. In Fig. 9, the ranking is denoted with ind, the feature distance is denoted with dis, (ind, dis) is marked on the tool's bounding box. We set the threshold of dis to 0.005, which means only tools having distance smaller than 0.005 can be selected as candidates. If the first ranked tool has a distance greater than 0.005 then the recommendation is failure, which means there is no tool can be used to complete the task.

In Fig. 9 (a), the fine-grained tool "bread knife", a coarse-grained tool "fruit knife" and several tools irrelevant with the task exist. The distance corresponding to "bread knife" is 0.0024 (smaller than 0.005) which is the minimum among all the distances, so the "bread knife" ranks first. The distance corresponding to "fruit knife" is 0.0027 (less than 0.005) which only greater than that of "bread knife" and smaller than all the other distances, so the "fruit knife" ranks second. The distances corresponding to tools irrelevant with the task such as "cup" and "hammer" are greater than the threshold and rank behind. So, the distances and the ranking comply with common sense.

In Fig. 9 (b), the fine-grained tool "bread knife" and two other coarse-grained tools are available, the distances corresponding to all the tools are smaller than the threshold and the "bread knife" ranks first. So, by learning the specific features, FTR-Net gains the ability to distinguish tools corresponding to the same coarse-grained task, which ensures FTR-Net recommend the correct fine-grained tool.

In Fig. 9 (c), the fine-grained tool "bread knife" is absent, a coarse-grained tool "fruit knife" and several tools irrelevant with the task exist. The "fruit knife" ranks first because the distance corresponding to it is 0.0025 (smaller than 0.005) which is the minimum among all the distances. So, by learning the common features, FRT-Net obtains the ability to recommend coarse-grained tool.

In the three actual scenarios, FTR-Net gives the most appropriate recommendation for the fine-grained task "cutting bread" although unseen tools during the training exist, that is "glue stick", "coke can" in Fig. 9 (a) (c) and "box cutter" in Fig. 9 (b). The results show that FTR-Net can learn fine-grained task related knowledge from FTT dataset, which makes it identify appropriate tool for the task based on the feature distance. The results also show that FTR-Net has well generalization ability, which makes it robust to work in actual circumstance.

VII. Conclusion

This paper defines fine-grained tool recommendation problem, and suggests a knowledge based solving schema including the definition of multi-granularity semantics for tasks, manipulated objects and relationships among them, public available FTT dataset and the baseline method FTR-Net. Experiments on the FTT test set show that FTR-Net can recommend the most appropriate tool for the fine-grained task in different cases. Experiments in the actual situation including unseen tools shows that FTR-Net has well generalization ability. FTR-Net is trained with data in the FTT dataset, so the scale and content of FTT affect the performance of FTR-Net. While new data are added to the FTT dataset, FTR-Net needs to be retrained. Therefore, in the future, we will consider how to incrementally learn fine-grained tool recommendation to help robots work more conveniently in the real world.

Acknowledgement

This research was supported by National Key R&D Program of China 2021ZD0111902, National Natural Science Foundation of China under Grants 61876012 and U21B2038, 62172022, U19B2039, and in part by the Foundation for China University Industry-university Research Innovation 2021JQR023.

References

[1] JJ Gibson. The theory of affordances. In *Perceiving, Acting, and Knowing: Toward and Ecological Psychology*, pages 97–82. NJ, USA: Erlbaum, 1977.

[2] Mohammed Hassanin, Salman Khan, and Murat Tahtali. Visual affordance and function understanding: A survey. *ACM Computing Surveys (CSUR)*, 54(3):1–35, 2021.

[3] Austin Myers, Ching L Teo, Cornelia Fermüller, and Yiannis Aloimonos. Affordance detection of tool parts from geometric features. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1374–1381. IEEE, 2015.

[4] Anh Nguyen, Dimitrios Kanoulas, Darwin G Caldwell, and Nikos G Tsagarakis. Detecting object affordances with convolutional neural networks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2765–2770. IEEE, 2016.

[5] Krishneel Chaudhary, Kei Okada, Masayuki Inaba, and Xiangyu Chen. Predicting part affordances of objects using two-stream fully convolutional network with multimodal inputs. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3096–3101. IEEE, 2018.

[6] Thanh-Toan Do, Anh Nguyen, and Ian Reid. Affordancenet: An end-to-end deep learning approach for object affordance detection. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1–5. IEEE, 2018.

[7] Xue Zhao, Yang Cao, and Yu Kang. Object affordance detection with relationship-aware network. *Neural Computing and Applications*, 32(18):14321–14333, 2020.

[8] Wenbai Chen, Chao He, WZ Chen, QL Chen, and PL Wu. A new semantic-based tool detection method for robots. *International Journal of Computers, Communications and Control*, 16(2), 2021.

[9] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[11] Johann Sawatzky, Abhilash Srikantha, and Juergen Gall. Weakly supervised affordance detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2795–2804, 2017.

[12] Fu-Jen Chu, Ruinian Xu, and Patricio A Vela. Learning affordance segmentation for real-world robotic manipulation via synthetic images. *IEEE Robotics and Automation Letters*, 4(2):1140–1147, 2019.

[13] Yixin Zhu, Yibiao Zhao, and Song Chun Zhu. Understanding tools: Task-oriented object modeling, learning and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2855–2864, 2015.

[14] Namiko Saito, Tetsuya Ogata, Satoshi Funabashi, Hiroki Mori, and Shigeki Sugano. How to select and use tools?: Active perception of target objects using multimodal deep learning. *IEEE Robotics and Automation Letters*, 6(2):2517–2524, 2021.

[15] Mike Stilman, Munzir Zafar, Can Erdogan, Peng Hou, Saul Reynolds-Haertle, and Gregory Tracy. Robots using environment objects as tools the macgyverparadigm for mobile manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2568–2568. IEEE, 2014.

[16] Johann Sawatzky, Yaser Souri, Christian Grund, and Jurgen Gall. What object should i use?-task driven object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7605–7614, 2019.

[17] Paulo Abelha, Frank Guerin, and Markus Schoeler. A model-based approach to finding substitute tools in 3d vision data. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2471–2478. IEEE, 2016.

[18] Paulo Abelha and Frank Guerin. Learning how a tool affords by simulating 3d models from the web. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4923–4929. IEEE, 2017.

[19] Madhura Thosar, Christian A Mueller, Georg Jaeger, Max Pfingsthorn, Michael Beetz, Sebastian Zug, and Till Mossakowski. Substitute selection for a missing tool using robot-centric conceptual knowledge of objects. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 972–979, 2020.

[20] Nithin Shrivatsav, Lakshmi Nair, and Sonia Chernova. Tool substitution with shape and material reasoning using dual neural networks. *arXiv preprint arXiv:1911.04521*, 2019.

[21] Lars Kunze, Moritz Tenorth, and Michael Beetz. Putting peoples common sense into knowledge bases of household robots. In *Annual Conference on Artificial Intelligence*, pages 151–159. Springer, 2010.

[22] Peter Kaiser, Mike Lewis, Ronald PA Petrick, Tamim Asfour, and Mark Steedman. Extracting common sense knowledge from text for robot planning. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3749–3756. IEEE, 2014.

[23] Ahmed Al-Moadhen, Renxi Qiu, Michael Packianather, Ze Ji, and Rossi Setchi. Integrating robot task planner with common-sense knowledge base to improve the efficiency of planning. *Procedia Computer Science*, 22:211–220, 2013.

[24] Nancy Fulda, Nathan Tibbetts, Zachary Brown, and David Wingate. Harvesting common-sense navigational knowledge for robotics from uncurated text corpora. In *Conference on Robot Learning*, pages 525–534. PMLR, 2017.

[25] 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.

[26] 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.

[27] David Paulius, Nicholas Eales, and Yu Sun. A motion taxonomy for manipulation embedding. *arXiv preprint arXiv:2007.06695*, 2020.

[28] 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.