NEOL: Toward Never-Ending Object Learning for Robots

Yuyin Sun and Dieter Fox

Abstract— **Learning to recognize objects based on names is a crucial capability for personal robots. Recent recognition methods successfully learn to recognize objects in a train-once-then-test setting. Yet, these methods do not apply readily to robotic settings, where a robot might continuously encounter new objects and new names. In this work, we present a framework for Never-Ending Object Learning (`NEOL`). Our framework automatically learns to organize object names into a semantic hierarchy using crowdsourcing and background knowledge bases. It then uses the hierarchy to improve the consistency and efficiency of annotating objects. It also adapts information from additional image datasets to learn object classifiers from a very small number of training examples. We present experiments to test the performance of the adaptation method and demonstrate the full system in a never-ending object learning experiment.**

I. INTRODUCTION

Humans are capable of learning continuously: we can learn to recognize new objects and to associate new names to existing objects. To be useful in long term, open ended deployments, personal robots must be able to do the same. But the predominate approach to object recognition is to train object classifiers on a predetermined set of objects with a predetermined set of names and then to test their performance on new instances of the same set of objects and names [1], [2], [3]. This train-once-then-test approach is inadequate for personal robots, as it is infeasible to assume that all objects and all possible names that people refer to them by are known in advance. So for these robots to be useful, they require the ability to learn not only new objects, but also new names to existing objects over time [4].

To see the potential problem of not learning new object names, we asked workers on Amazon Mechanical Turk (AMT) to name objects in a popular RGB-D dataset [2]. This dataset contains objects that are organized according to WordNet categories (a pre-specified set of names). We found that only 75% of the names given by the workers are included in WordNet. Thus even if a robot correctly labels 100% of the objects to their WordNet categories, it would only correctly name 75% of the objects to their AMT worker names simply because it lacked the ability to associate new names to existing ones. This is important, because people do not only use category names to refer to objects [5]. For instance, a can of "Pepsi Cola" might be referred to as "Pepsi", "pop", "soda", "can", or "Pepsi can". To deal with the challenge of learning novel objects and names, we introduce `NEOL`, a framework for Never-Ending Object Learning. `NEOL` learns objects and their names continuously, adding and improving classifiers after every object encounter—rather than learn everything in a single training session with the train-once-then-test approach.

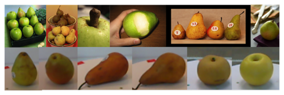
There are two main challenges to developing a framework for never-ending object learning. First, how should objects and their corresponding names be organized? A naïve approach would be to train an independent classifier for every object name, using instances with that name as positive examples and instances of all other object names as negative examples. However, this approach ignores the important relationships between names. For instance as apples are also fruits, every object named "apple" should also be a positive training example for the name "fruit". A lot of false positives will happen if the naïve approach is deployed, and could be especially harmful for never-ending learning paradigm, where training data are quite limited. We decide to organize names and objects according to semantic hierarchies of concepts. Semantic hierarchies of concepts have been shown to be useful in building classifiers for object recognition, and improving the recognition accuracy [1], [6], [7]. Most existing works use existing hierarchies, which may not be optimal for their tasks. To deal with this challenge, `NEOL` automatically learns a hierarchy over object names using crowdsourcing—inserting new names as they occur.

The second challenge for never-ending learning is to learn object classifiers from a very small number of training examples, which is particularly important when an object is encountered for the first time. One approach is to use existing annotated images collected for non-robotic applications [1], [8]. However, these images may be very different from those perceived by the robot (Figure 1). To understand how different they are, we trained object recognition classifiers on ImageNet dataset [1], and tested on the RGB-D object dataset [2]. We found the average precision across all objects on the RGB-D object dataset to be 33% lower than the average precision across same type of objects on the ImageNet testset. This drop in performance is not due to the difficulty of the RGB-D dataset, but rather due to the fact that ImageNet images have very different and diverse appearances. For example, cereal boxes can be classified on the ImageNet testset with more than 90% precision. However, the same classifier has less than 50% precision on the "easier" cereal box images in the RGB-D dataset. As we will see, `NEOL` uses domain adaptation to leverage existing image datasets while avoiding poor performance usually associated with domain transfer.

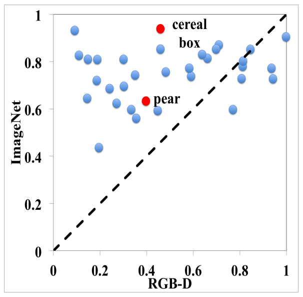
Our contributions: We introduce `NEOL`, a principled framework for robots to learn new objects and new object names over time. We extend our previous method for building semantic hierarchies of names using existing knowledge bases.

Yuyin Sun and Dieter Fox are with the Department of Computer Science & Engineering, University of Washington, Seattle, WA 98195, USA. {sunyuyin, fox}@cs.washington.edu

(a) *cereal box* samples from ImageNet (top) and RGB-D (bottom).



(c) *pear* samples from ImageNet (top) and RGBD (bottom).



(b) Average precisions of ImageNet classifiers tested on ImageNet vs RGB-D.

Fig. 1: ImageNet images have very different appearance from the RGB-D images. Recognition models built using the ImageNet dataset performs poorly on the RGB-D dataset.

We design a novel method that uses both in-domain RGB-D images and out-domain images to accurately recognize objects by names. Extensive evaluations of the proposed method show the robust results even under challenging settings.

The remainder of this paper is structured as follows: After discussing related work in Section II, we present the `NEOL` framework in Section III, the experiments in Section IV, and the conclusion in Section V.

II. RELATED WORK

Object Recognition: The predominant approach to object recognition is to learn classifiers based on labeled training data and to test the classifiers on held out data [2], [1], [3]. Such approaches, for example deep convolutional neural networks (CNN) [3], have obtained state-of-the-art performance on many large-scale classification tasks, including on RGB-D recognition [9] and on the ImageNet challenge [3]. However, CNNs require large amounts of training data to reach this level of performance and they do not address the setting where new objects and new names need to be added to the system. Despite this limitation, CNNs can extract discriminative features that are generic to all (even unseen) objects. For this reason, we chose to use a CNN trained on ImageNet [10] as a feature extractor.

Never-Ending Learning: Never-ending learning systems have been of the latest interest because they can learn many possible functions (i.e., different types of knowledge) in a cumulative nature. The functions learned over years of accumulation form an extensive background knowledge that improves subsequent learning [11]. Early works have focused on learning natural language (e.g. Never-Ending Language Learning [12]), learning visual information (e.g. Never-Ending Image Learning [13], [14]), autonomous control [15], and others. In this work, we build a never-ending system to learn to recognize objects through communicating with humans, which is a new direction of never-ending learning. It also worth noting that most never-ending learning work focus on coverage of knowledge rather than accuracy, therefore cannot satisfy the requirement of robotics applications. Our system emphasizes both coverage and accuracy.

Name Hierarchies: Semantic hierarchies have been shown to improve the scalability and efficiency of building recognition algorithms. For example, Deng *et al.* [6] deploy a hierarchy to efficiently annotate objects with multiple labels. Lai *et al.* [7] organize objects into a semantic tree to enable scalable recognition in real-time. The hierarchies in these works are either taken from online knowledge bases [16], or are hand-designed by domain experts [6]. However, the world we live in is complex and cannot be fully specified in advance; for a personal robot to interact with humans well it must be able to continuously update its hierarchy. To address this issue Blei *et al.* [17] propose a method to automatically build hierarchies, while Sun *et al.* [4] propose a method to insert new names into an existing hierarchy as leaf nodes. Both of these methods have shortcomings; The method in [17] fails to capture relationships between concepts as perceived by people, while Sun *et al.* fails when a new name is an internal node. A recent work by Sun *et al.* [18] propose building concept hierarchies from scratch using crowdsourcing in a principled way. This method can handle inserting new names as both leaf and internal nodes. We extend the work by combining the crowdsourcing and existing knowledge bases to build hierarchies efficiently and cost-effectively.

Domain Adaptation: Many large-scale labeled image datasets, such as ImageNet [1] and LabelMe [8] have been established recently for evaluating computer vision algorithms. These datasets are inherently biased to their application domains [19] however, and models built using these datasets do not generalize well to robotics applications since it was not their intended application. To deal with dataset bias, many visual domain adaptation methods have been introduced; for example, unsupervised adaptation [20], [21], supervised adaptation [22], deep architecture based method [23]. However, these methods are only able to adapt the visual data from source to target domains when the labels from both domains are the same, which is not the case for many robotics application scenarios.

III. APPROACH

Figure 2 shows the pipeline of `NEOL`. The input to `NEOL` is a stream of object images and associated names. If an image-name pair contains a new name, then `NEOL` inserts the name into the hierarchy using crowdsourcing and a background name hierarchy such as WordNet (see Section III-B). The new object image is then added to the training data, and the name annotation is propagated according to the hierarchy. For instance, if `NEOL` has already incorporated "apple" and "pear", then the insertion of the new object name "fruit" would automatically tag all previous "apple" and "pear" images as positive examples for "fruit" (see Section III-A). To improve the performance of the image classifiers, `NEOL` uses annotated images from an existing dataset whenever possible (Section III-C).

We use the following notations. $\mathcal{N} = \{n_1, \ldots, n_L\}$ denotes a set of names, and $\mathcal{O} = \{o_1, \ldots, o_M\}$ a set of object images. Each o_m is associated with at least one name in $\mathcal{N}$. All names in $\mathcal{N}$ are organized in a tree-structured hierarchy H. We use

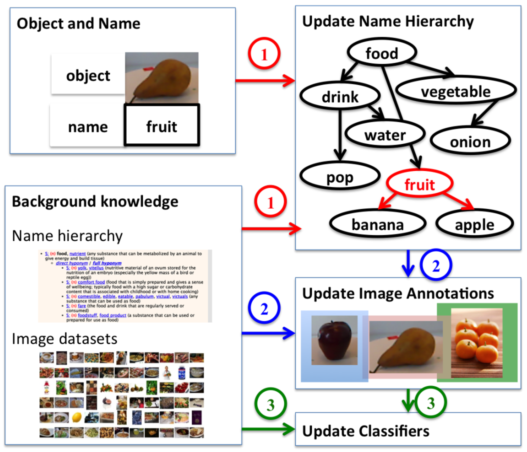


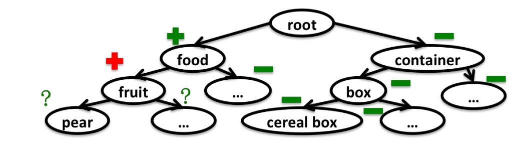
Fig. 2: Pipeline of NEOL. At each time, the input consists of an object image and an associated name. If the object name is new, then NEOL updates the name hierarchy using crowdsourcing and information available from an existing word hierarchy. The following step is to propagate the name annotations of images using the updated hierarchy. In the final step, NEOL updates all corresponding image classifiers, using background images to improve generalization.

$f_1, \ldots, f_L$ to denote classifiers that take the image feature $\boldsymbol{x}_m$ of an object image o_m and predict $f_l(\boldsymbol{x}_m)$ indicating whether o_m should be annotated with the name n_l.

A. Hierarchies for Consistent Image Annotation

As we will show in our experiments, organizing object names into a hierarchy significantly improves the training of visual object classifiers. This is mostly due to the fact that the same object might be referred to by multiple names, causing sparsity of training data and even confusion for the image annotation. For instance, imagine a pear is presented to the robot along with the name *pear*. This image should obviously become a positive training example for a *pear* classifier. Unfortunately, it is not obvious how to use this image for the other classifiers since it is not necessarily a negative example for every other name. By knowing the hierarchical organization over names, a system could determine, for example, that the pear image should also become a positive example for the *fruit* classifier, thereby adding to that classifier's training set and avoiding that it becomes a negative example for the fruit classifier.

NEOL uses a hierarchical organization over object names to keep its image annotations consistent (we assume that all the names have already been inserted into the hierarchy in this subsection, and will discusse how to generate the hierarchy in the next subsection). If an object is labeled as a positive example of a name n_l, then the *positive* label will be propagated up along the path between n_l and the root node of the hierarchy. At the same time, the object becomes a *negative* example for all other nodes in the hierarchy *except* the descendants of n_l. This is due to the fact that, without further annotation, it is not known which of n_l's descendants the object belongs to. Figure 3 demonstrates label propagation using a hierarchy.



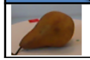
label / object	pear	fruit	food	cereal box	box	container
	?	🟥+	🟩+	−	−	−

Fig. 3: Label propagation using a hierarchy. The red sign is the label given by a person. The green signs are propagated using the hierarchy. Since *fruit* is a *food* in the hierarchy, the pear image should also be labeled as a positive example of *food*. It can furthermore be inferred, for instance, that *food* is not a *container*. Without additional information, it cannot be inferred whether the new object should be a positive or negative example for *pear* or any of the other descendants of *fruit*.

B. Building a Name Hierarchy via Crowdsourcing

As stated above, existing name hierarchies such as the WordNet ontology do not contain all possible names people might use to refer to objects. Furthermore, these hierarchies are built by domain experts and do not necessarily conform with people's ideas of how object names relate to each other. Therefore, we decided to incorporate an online hierarchy learning approach into NEOL. Specifically, we adopt and extend our previous method [18] to build name hierarchies. We summarize the gist of the method in the following, and more details can be found in [18].

Specifically, we want to build a hierarchy H over L nodes (names) by asking crowdsourcing workers questions regarding the hierarchy structure. Our goal is to find the hierarchy that best matches workers' answers, in a space $\mathcal{H} = \{H_1, \ldots, H_M\}$ over all possible hierarchies with L nodes. However, workers' answers are inherently noisy. Even if every worker gives his best possible answers, name relationships might be ambiguous and there might not exist a single hierarchy that consistently explains all the workers' answers. To capture the uncertainty over semantic hierarchies and worker noise, we use a Bayesian framework to estimate probability distributions over hierarchies, rather than using a single, best guess.

The key challenge here is the huge number of candidate hierarchies, which is $(L+1)^{L-1}$ for $L+1$ nodes (we add one root node), resulting in 1,296 trees for 6 names, but already $2.3579e+09$ trees for only 11 names. It is thus intractable to directly model the distribution as a multinomial. We introduce a compact model to represent the family of distributions over hierarchies:

$$P(H|W) = \frac{\prod_{e_{i,j} \in H} W_{i,j}}{Z(W)}, \qquad (1)$$

where $W_{i,j}$ is a non-negative weight for the edge $e_{i,j}$ representing whether node n_j is an immediate descendent of node n_i, and $Z(W) = \sum_{H' \in \mathcal{H}} \prod_{e_{i,j} \in H'} W_{i,j}$ is the partition function.

Our method updates the model parameter W by asking workers a sequence of questions $q^{(1)}, \ldots, q^{(t)}, \ldots$ and getting the corresponding answers $a^{(1)}, \ldots, a^{(t)}, \ldots$. Each question $q^{(t)}$ is posed to determine whether a direct path $p_{i,j}$ exists in the hierarchy from n_i to n_j.An example would be "*Is apple a type of fruit?*". The answer $a^{(t)}$ to the question is denoted as $a_{i,j} \in \{0, 1\}$, where $a_{i,j} = 1$ means a worker believes that there is a path from node n_i to n_j, and vice versa. The method then updates the posterior $P(H|W^{(t)})$ at the (t)-th iteration using Bayes' rule:

$$P(H|W^{(t)}) \propto P(H|W^{(t-1)})f(a^{(t)}|H), \tag{2}$$

where $f(a^{(t)}|H)$ is the likelihood of getting answer $a^{(t)}$ given the hierarchy H. It can be defined as follows

$$f(a_{i,j}|H) = \begin{cases} (1-\gamma)^{a_{i,j}}\gamma^{1-a_{i,j}}, \text{ if } p_{i,j} \in H \\ \gamma^{a_{i,j}}(1-\gamma)^{1-a_{i,j}}, \text{ otherwise} \end{cases}.$$

Here $p_{i,j} \in H$ checks the existence of the path $p_{i,j}$ in H and γ is the noise rate of workers.

Unfortunately, the likelihood is not conjugate to the posterior. Hence, there is no analytical form for updating the weights. Sun *et al.* [18] propose to update the weight matrix by approximate inference. The optimal solution W^* is found by minimizing the KL-divergence [24] between $P(H|W)$ and $P(H|W^{(t)})f(a^{(t+1)}|H)$:

$$W^* = \arg\min_W KL(P(H|W^{(t)})f(a^{(t+1)}|H) \| P(H|W)),$$

which can be solved efficiently using sampling method and gradient descent.

Though using crowdsourcing to answer a single question is inexpensive, answering all relevant questions via crowdsourcing is still very costly, especially for building large hierarchies. To save questions, the approach from [18] selects the sequence of path questions based on information gain, thereby reducing the number of questions needed to be asked.

Additionally, due to the noise in workers' answers, the same question has to be answered by multiple workers to improve confidence. Fortunately, we can avoid asking a lot of questions using source domain semantic hierarchies, e.g. the WordNet. To see, if both nodes n_i and n_j are in the WordNet hierarchy, we can use WordNet to answer the path question $p_{i,j}$ by checking whether n_i is a hypernym of n_j in WordNet. Since WordNet is built by experts, the error rate γ for that answer would be very small. On the RGB-D dataset, we found that using WordNet for answering questions saves about 40% of the queries, also resulting in a learned hierarchy that is more consistent with WordNet if there are overlapping nodes between the them.

After collecting sufficient information from workers and WordNet, our approach produces a very concentrated probability distribution over a few hierarchies. NEOL uses the Maximum A Posteriori (MAP) hierarchy for label propagation. Finding the MAP tree of the distribution defined in equation (1) can be accomplished efficiently using the minimum-spanning tree algorithm [25].

C. Leveraging Background Datasets

We now show how to incorporate images and labels from a background dataset to build classifiers that generalize well in the robotics target domain. In what follows, we use the term *target domain* to refer to the environment where the robot is observing the objects, and *source domain* to refer to additional background knowledge bases such as WordNet and ImageNet. Following standard convention, we use superscript S to indicate concepts from the source domain, and superscript T for the target domain.

Using the hierarchy and a set of image-name pairs provided by humans, the robot is able to annotate training images $\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_M^T$ with name labels $\boldsymbol{y}_1^T, \ldots, \boldsymbol{y}_M^T$ from the target domain. Each $\boldsymbol{x}_m^T \in \mathbb{R}^d$ has two components, $\boldsymbol{x}_{m,RGB}^T \in \mathbb{R}^{d_{RGB}}$ and $\boldsymbol{x}_{m,depth}^T \in \mathbb{R}^{d_{depth}}$, corresponding to features extracted from an RGB image and a depth map, respectively. (Though we assume a robot is using an RGB-D camera, our method can be easily extended to other sensors. Feature extraction methods are also flexible and can be changed for different applications.) $\boldsymbol{y}_m^T \in \{0, 1\}^L$ are the annotations of o_m, where $y_m^{(l)T} = 1$ indicates that the object o_m can be referred to by the name n_l. Some elements of $\boldsymbol{y}$ are provided by humans by referring objects to the robot using the corresponding names. The rest of $\boldsymbol{y}$ are inferred using the label propagation method described in section III-A.

In the never-ending learning setting, a challenge is that a robot might not be able to acquire a large number of training examples for each label. For instance, it might initially see only a single soda can, which makes it extremely hard to generalize to other types of soda cans. To increase generalization capabilities, NEOL uses additional training images from an existing background, or source domain dataset, such as ImageNet [1]. The source domain dataset has a large amount of RGB images for a set of categories $\mathcal{N}^S = \{n_1^S, \ldots, n_N^S\}$. NEOL uses the same feature extraction method applied to target domain RGB images to extract features $\boldsymbol{x}_{1,RGB}^S, \ldots, \boldsymbol{x}_{M,RGB}^S$ from source domain images.

Note that some target domain names in $\mathcal{N}^T$ might not exist in the source domain $\mathcal{N}^S$. NEOL deploys the hierarchy H^T over the target domain to find source domain images even when the name is not in $\mathcal{N}^S$. To see, if a name n_l^T is also in $\mathcal{N}^S$, then the label of $y^{(l)S}$ for the name n_l^T will be the direct copy from ImageNet. If a name n_l^T is not in $\mathcal{N}^S$, but some of its descendant nodes $n_{l_1}^T, \ldots, n_{l_n}^T$ in the hierarchy H^T are in $\mathcal{N}^T$, then all images from ImageNet with any labels in $n_{l_1}^T, \ldots, n_{l_n}^T$ will also be labeled as n_l^T. For instance, the name "crockery" is not in ImageNet. However, it is a parent of "saucer" and "dish" in the learned hierarchy. Since these categories are in ImageNet, NEOL can use images from these two categories and their descendants as positive examples for "crockery".

Even if the source domain and the target domain images are from the same category, the images can be very different, causing a significant performance drop when the classifiers built on the source domain are applied directly to the target domain (Figure 1 (b)). To deal with such a domain-mismatch, our method is based on the observation that the source domain dataset is often the combination of several distinct sub-groups, each of which has its own characteristic [20], [26]. Some groups are more similar to the target domain, while some others are not. For example, the top row of Figure 1 (a) shows images of cereal boxes from ImageNet. Some of them are more similar to the cereal boxes in the RGB-D dataset (bottom row), which are mostly single cereal box images. Some are not, e.g., images with cereal boxes on a shelf. If we model the source domain dataset using a single model, the discriminative information from the different groups will be mixed together, thereby diminishing the benefit of adding the source domain. To avoid this, we group source domain images into different sub-groups and a learn specific classifier for each sub-group. For grouping, we have tried both sophisticated methods [20] and simple clustering methods such as K-means. The performance of the different methods was very similar in our context, so we decided to use K-means to partition the full source-domain dataset into K groups.

Next, we build one classifier for each sub-group by combining all the target-domain RGB and depth images with the source domain images from that sub-group. A classifier for the k-th sub-group in the l-th category is learned by optimizing the following objective:

$$\boldsymbol{w}_{l(k)} = \arg\min_{\boldsymbol{w}} \|\boldsymbol{w}\|^2 + C_1 \sum_{m=1}^{M_k^T} h\left(\boldsymbol{w}'_{RGB}\boldsymbol{x}^T_{m,rgb}, y_m^{(l)}\right)$$
$$+ C_1 \sum_{m=1}^{M^S} h\left(\boldsymbol{w}'_{RGB}\boldsymbol{x}^T_{m,rgb}, y_m^{(l)}\right)$$
$$+ C_2 \sum_{m=1}^{M^S} h\left(\boldsymbol{w}'_{depth}\boldsymbol{x}^T_{m,depth}, y_m^{(l)}\right), \forall l \in \{1, \ldots, L\}$$

where $\boldsymbol{w} = [\boldsymbol{w}'_{RGB}, \boldsymbol{w}'_{depth}]'$, $h(\cdot,\cdot)$ is the hinge loss, and M_k^T are the target domain images in the k-th cluster. C_1 and C_2 balance the importance of RGB and depth channels. This objective function incorporates RGB images from both the source and the target domains, as well as depth information that exists only for the target domain. When applying the models to predict new images from class l, we average the prediction scores given by all the K sub-group classifiers.

IV. EXPERIMENTS

We evaluate the quality of NEOL via two sets of experiments. First, we evaluate the method used by NEOL to leverage background image databases. Next, we evaluate the full system of NEOL under the never-ending learning setting in Section IV-B. The hierarchy building method described in III-B has been demonstrated to be effective in [18], and we refer readers to that paper for more details.

A. *Leveraging Background Image Databases*

We use the RGB-D dataset [2] as the target domain dataset, and ImageNet [1] as the source domain to evaluate the domain-adaptation method of NEOL. The RGB-D dataset contains 300 objects organized under 51 categories. The task is to build classifiers for the 51 categories using training data, and evaluate the average classification accuracy over all categories on a testing set.

We compare our approach with several competing approaches (summarized in Table I). The Baseline method

TABLE I: Comparison of category level classification accuracies on the RGB-D Object dataset. LOIO: Leave One Instance Out. 1I/C: 1 instance per category. 1V/O: 1 view per object. Differences in accuracy relative to the baseline method are marked by colors (red: improvement, blue: performance decrease). All numbers are averaged over 10 random data splits.

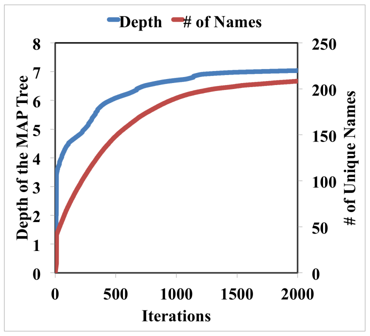
method	Category Accuracy(%)		
	LOIO	1I/C	1V/O
	RGB/D	RGB/D	RGB/D
Baseline [9]	83.6/89.2	58.8/67.3	65.1/70.2
Baseline-S	−58.7/ − −−	−33.9/ − −−	−40.2/ − −−
Baseline-T+S	−6.4/−5.3	−5.0/−3.9	−20.1/−17.9
GFK [21]	−53.2/ − −−	−28.4/ − −−	−34.7/ − −−
Fru [22]	−1.1/−1.8	+1.4/−0.5	−6.4/−4.9
NEOL	+1.3/+0.9	+5.5/+5.9	+4.6/+2.8
NEOL	**84.9/90.1**	**64.3/73.2**	**69.7/73.0**

is trained only on RGB-D data using CNN-features [9] extracted via Caffe [10] (this method achieves state-of-the-art performance on the RGB-D dataset, but any other features could be used as well). Baseline-S uses classifiers only trained on the source domain ImageNet, and GFK uses a domain-adaptation method on ImageNet data [21]. We further compare with several methods using both the target and the source domain images. Baseline-S+T simply combines source-domain and target-domain images into a larger-size training set. Frustratingly-easy domain-adaptation is a supervised method [22] (Fru), which achieves state-of-the-art performance in many domain-adaptation tasks. If a method does not specify a classifier, linear SVM is used. For all the comparison methods, we use the default parameters. For the adaptation used in NEOL, the parameter K is set to be 10 across all the experiments (we tried larger Ks, but did not observe a significant difference). The results are summarized in Table I, where the first and last row give absolute accuracies, and the other rows provide changes relative to the baseline.

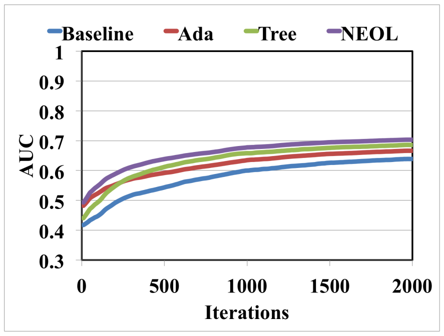
Leave One Instance Out (LOIO) For the first experiment, we conduct Leave-One-Instance-Out test following the protocol in [2]. Each object in both training and testing set has 90 views.

The target-domain baseline method achieves 89% accuracy. But the other methods except for NEOL cause decreases in performance. Methods using only source domain images do not consider the target domain, therefore perform the worst. Simply combining source-domain and target-domain data without adaptation also harms the performance (Baseline-T+S). Frustratingly-easy domain adaptation (Fru) builds a single domain adaptation model using a feature replication method to bridge the gap between domains. Although it improves the performance over the naive combination, the performance is still worse than the baseline method, which uses only target domain data. Our approach outperforms the baseline methods with 1.3% and 0.9% improvement on RGB and RGB-D data, respectively. The improvement over the baseline is statistically significant.

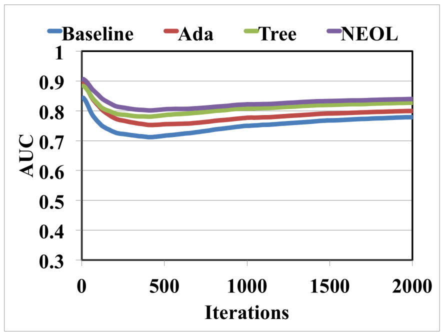
It is worth noting that the LOIO setting in the first experiment has a large amount of target-domain training instances and poses. Several methods in Table I perform well with sufficient training examples. However, in never-ending learning, training instances may be sparse. Therefore, we test all the competing methods under the small-training-set setting



(a) Iterations vs Depth of the MAP Tree and the number of unique names.



(b) AUC on new instances.



(c) AUC on seen instances.

Fig. 4: Results of the Never-Ending-Object-Learning experiments. (a) shows growth of tree and number of unique names as objects are presented. (b,c): Baseline uses classifiers that determine training sets only through direct name labels. Ada means Baseline plus adaptation, Tree is Baseline plus hierarchy for label propagation. NEOL uses both adaptation and hierarchy.

in the following experiments.

One Instance per Category (1I/C) For the second experiment, we give only one target-domain instance of each object category for training. To compare with LOIO, we adopt the same testing protocol and leave one instance out of each category as testing set. The second column of Table I shows the results. As the number of training instances decreases, the performance of the baseline drops to 67.3% (RGB-D). Baseline-T+S and Fru use source domain images, yet do not manage to improve the performance over the baseline. On the other hand, NEOL achieves a 5.9% improvement over the baseline. NEOL gains more relative improvement when only target-domain RGB images are used, compared with using both RGB images and depths (RGB-D). This can be explained by the fact that ImageNet has only RGB images.

One View per Object (1V/O) For the third experiment, we evaluate how different methods handle variance between object poses. We follow the leave-one-instance-out setting in the first experiment, but only give 1 view for each training instance (1V/O in Table I). NEOL obtains a 2.8% improvement over the baseline methods, which shows the advantage of using NEOL to cope with the variance between object poses.

To summarize, the domain adaptation performed by NEOL is the only one that can take advantage of additional image data available in ImageNet. Furthermore, the improvements are not drastic, since the CNN features already incorporate a lot of stability with respect to features invariances.

B. *Never-Ending Object Learning*

We evaluate NEOL on the task of never-ending object learning. The setting is as follows. A person is continuously teaching a robot objects by showing it a few views and names of the objects. The performance of the robot is evaluated on both previously seen and unseen objects, and asking it to recognize objects that can be referred to by any names known to the robot. The task is a multi-label problem, therefore the performance is measured by the average AUC over all labels.

We simulate the previous setting using the RGB-D dataset by selecting 150 objects for sequencial training. At each iteration, one of the objects is randomly picked and shown to the robot. The robot is given 3 views of each object, and gets one name of the new object from a worker at AMT (generated by providing the worker images of the object). The iterative process repeats for $2,000$ iterations in each trial. We repeated 10 random trials and report the average performance in this section. The remaining 150 instances of the RGB-D dataset are used only for testing the performance.

At each iteration, the robot uses NEOL (Figure 2) to improve the classifiers: first, if the name is new, it inserts the name into the hierarchy using the method described in III-B; second, it updates training examples and annotations using the hierarchy as described in III-A; finally, it updates the classifiers of existing names using the domain adaptation method following section III-C. We modify the second step by treating all unknown labels (marked by "?" in Figure 3) as negative rather than ignoring them during training. Even though this modification might introduce a few false negative examples, it gives improved results since it avoids ignoring many training examples.

Figure 4 (a) plots some important statistics of the hierarchies built by the system. The red curve shows the number of unique names used by workers to refer to objects. It increases almost linearly for the first 200 iterations, because almost all the objects are new to the robot, and new objects yield new names. The number of names still grows between 200 and $1,500$ iterations due to the fact that different people use different names to refer to the same object. After $1,500$ iterations, the number remains flat since new names are very rare. The curve showing the depth ot the MAP hierarchy tree has a similar trend. The hierarchy has about 4 levels after about 50 iterations, and eventually grows to 7 levels.

Figure 4 (b) shows the performance of applying the learned models to recognize new objects. We compare both NEOL and the baseline [9] in this experiment, which does not take advantage of the hierarchy or ImageNet data. To evaluate how different components of NEOL contribute to the full system, we also include two more baselines by adding the hierarchy (Section III-B) and adaptation (Section III-C) to baseline, denoted as Tree and Adaptation, respectively.

After the first iteration, NEOL attains an AUC of 0.5, while the baseline only achieves 0.4. Since there is only one name, using the hierarchy does not yield a lot of information. Ada and NEOL thus perform almost identical. This suggests that at the early stage, the gain of NEOL is mainly due to the domain adaptation. The hierarchy starts showing its effectiveness after 50 iterations. As the hierarchy gets deeper, it is able to capture meaningful semantic relations between names. As can be seen in Figure 4 (b), Tree surpasses Ada at around 200 iterations.

After $1,000$ iterations, the marginal improvement of NEOL compared with the baseline is still over 10%. As more objects are shown to the robot, NEOL and the baseline converge, but NEOL retains a 5% advantage until $2,000$ iterations. This shows that NEOL is able to learn new objects and names more efficiently than the state-of-the-art methods.

We also apply different methods to different views of the instances that have already been shown to the robot. The results are shown in Figure 4 (c). Note that the AUC initially drops since as more objects are added, there is more chance for confusion. Among all the competing methods, NEOL performs the best. Since NEOL has seen some labels of these objects during the training stage, it achieves much higher AUC on these objects than unseen objects.

V. CONCLUSIONS

In this paper, we presented the never-ending learning framework NEOL for a robot to learn new objects and names over its lifetime. NEOL organizes the stream of object images and names into a semantic hierarchy using crowdsourcing. The number of queries for building the hierarchy can be reduced by leveraging an existing knowledge base such as WordNet. NEOL uses the hierarchy to organize the objects it perceives and to ensure that image annotation is consistent. NEOL also includes a domain adaptation method to combine images from existing image databases with the images the robot perceives in order to better generalize to unseen objects.

Our experiments on the RGB-D dataset using ImageNet and WordNet as additional background information demonstrate that NEOL is able to learn objects efficiently and effectively under the setting of never-ending object learning. The experiments also show that building and using a name hierarchy for consistent image annotation significantly improves the classification capabilities of NEOL. Using ImageNet as additional background knowledge further improves performance, where the improvements due to domain adaptation are most significant when only small sets of objects are perceived in the environment.

In this work, NEOL uses the MAP hierarchy over names to propagate annotations. However, there might not always exist a unique, correct hierarchy. For example, some people think “tomato” is a type of “fruit”, but others may think it is a “vegetable”. Our hierarchy building algorithm [18] is able to capture such uncertainty over hierarchies and it would be interesting to consider uncertainty in building a recognition system that is robust to “miscommunication” between humans and robots.

Videos illustrating the learning process are available at `http://rse-lab.cs.washington.edu/projects/neol`.

ACKNOWLEDGMENT

This work was funded in part by the Intel Science and Technology Center for Pervasive Computing, ONR grant N00014-13-1-0720, and ARO grant W911NF-12-1-0197.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database,” in *CVPR*, 2009.

[2] K. Lai, L. Bo, X. Ren, and D. Fox, “A large-scale hierarchical multi-view rgb-d object dataset,” in *ICRA*, 2011.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *NIPS*. Curran Associates, Inc., 2012.

[4] Y. Sun, L. Bo, and D. Fox, “Learning to recognize new objects,” in *ICRA*, 2013.

[5] S. Kazemzadeh, V. Ordonez, M. Matten, and T. L. Berg, “Referitgame: Referring to objects in photographs of natural scenes,” in *EMNLP*, 2014.

[6] J. Deng, O. Russakovsky, J. Krause, M. S. Bernstein, A. Berg, and L. Fei-Fei, “Scalable multi-label annotation,” in *CHI*, 2014.

[7] K. Lai, L. Bo, X. Ren, and D. Fox, “A scalable tree-based approach for joint object and pose recognition,” in *AAAI*, 2011.

[8] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, “Labelme: A database and web-based tool for image annotation,” *International Journal on Computer Vision*, vol. 77, no. 1-3, pp. 157–173, May 2008.

[9] M. Schwarz, H. Schulz, and S. Behnke, “RGB-D object recognition and pose estimation based on pre-trained convolutional neural network features,” in *ICRA*, 2015.

[10] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *CoRR*, vol. abs/1408.5093, 2014. [Online]. Available: http://arxiv.org/abs/1408.5093

[11] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling, “Never-ending learning,” in *AAAI*, 2015.

[12] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., and T. M. Mitchell, “Toward an architecture for never-ending language learning,” in *AAAI*, 2010.

[13] X. Chen, A. Shrivastava, and A. Gupta, “Neil: Extracting visual knowledge from web data,” in *ICCV*, December 2013.

[14] X.-S. Hua and J. Li, “Prajna: Towards recognizing whatever you want from images without image labeling,” in *AAAI*, January 2015.

[15] S. Thrun and T. M. Mitchell, “Lifelong robot learning,” Robotics and Autonomous Systems, Tech. Rep., 1993.

[16] C. Fellbaum, *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[17] D. Blei, A. Ng, and M. Jordan, “Latent dirichlet allocation,” *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[18] Y. Sun, A. Singla, D. Fox, and A. Krause, “Building hierarchies of concepts via crowdsourcing,” in *IJCAI*, 2015.

[19] A. Torralba and A. A. Efros, “Unbiased look at dataset bias,” in *CVPR*, June 2011.

[20] B. Gong, K. Grauman, and F. Sha, “Reshaping visual datasets for domain adaptation,” in *NIPS*. Curran Associates, Inc., 2013.

[21] B. Gong, Y. Shi, F. Sha, and K. Grauman, “Geodesic flow kernel for unsupervised domain adaptation.” in *CVPR*, 2012.

[22] H. Daume III, “Frustratingly easy domain adaptation,” in *ACL*, 2007.

[23] J. Hoffman, E. Tzeng, J. Donahue, Y. Jia, K. Saenko, and T. Darrell, “One-shot adaptation of supervised deep convolutional models,” *CoRR*, vol. abs/1312.6204, 2013. [Online]. Available: http://arxiv.org/abs/1312.6204

[24] S. Kullback and R. Leibler, “On information and sufficiency,” *Annals of Mathematical Statistics*, pp. 49–86, 1951.

[25] Y. Chu and T. Liu, “On the shortest arborescence of a directed graph,” *Science Sinica*, vol. 14, 1965.

[26] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko, “Discovering latent domains for multisource domain adaptation,” in *ECCV*, 2012.