Towards a Robot Architecture for Situated Lifelong Object Learning

Jose L. Part and Oliver Lemon

***Abstract*— The ability to acquire knowledge incrementally and after deployment is of utmost importance for robots operating in the real world. Moreover, robots that have to operate alongside people need to be able to interact in a way that is intuitive for the users, e.g., by understanding and producing natural language. In this paper we present a first prototype of a robot architecture developed for situated lifelong object learning. The system is able to communicate with its users through natural language and perform object learning and recognition on the spot through situated interactions. In this first stage, we evaluate the system in terms of recognition accuracy which gives an indirect measure of the quality of the collected data with the proposed pipeline. Our results show that the robot can use this data for both learning and recognition with acceptable incremental performance. We also discuss limitations and steps that are necessary in order to improve performance as well as to shed some light on system usability.**

I. INTRODUCTION

Robots that operate in dynamic and unstructured environments need to be capable of adapting and expanding their knowledge as they interact with their users and surroundings.

Over the past few years, there has been a renewed surge in interest on lifelong machine learning [1], [2] due to its high relevance to real world applications. However, most approaches usually focus exclusively on the learning model and ignore the data acquisition steps by relying on existing and sometimes curated datasets. In addition, these models are often evaluated on "toy" datasets that fail to capture the complexities of real-world objects (e.g., MNIST).

In this paper, we describe a system architecture that enables robots to learn new objects by interacting with its users in a lifelong manner. The system can detect objects on supporting planes and process the corresponding input data to make it suitable to be consumed by the learning algorithm. In addition, the system can engage in clarification dialogue to address errors, e.g., speech recognition or entity extraction errors.

In this study, the following assumptions are made:

- Objects are assumed to be on a supporting plane.
- There is a single object on the supporting plane.
- There is a single user so we don't explicitly address the problem of having multiple referring expressions for the same objects, e.g., laptop and notebook. However, we do provide some insights on how this might be addressed as discussed in Sec. V-C.

Jose L. Part is with the Edinburgh Centre for Robotics, Edinburgh, Scotland, UK jose.part@ed.ac.uk

Oliver Lemon is with Heriot-Watt University, Edinburgh, Scotland, UK o.lemon@hw.ac.uk

Fig. 1. One of the locations used for data collection.

II. RELATED WORK

Some work has approached the related (precursory) problem of lifelong object discovery and autonomous 3D model building. For instance, Collet et al. [3] developed a system that allows a robot to collect data from exploring an environment and analyze it in order to discover objects within. The approach relies on domain knowledge which can consist of spatial and temporal constraints, prior knowledge, similarity measures and so on. Finman et al. [4] on the other hand, learn the object segmentations from looking at changes in dense RGB-D maps over time. In a similar approach, Fäulhammer et al. [5] proposed to model the static parts of the environment and detect the objects of interest by extracting dynamic clusters, i.e., regions of the point cloud that change in between visits to that particular location in the environment. Once a cluster has been identified and selected, the robot plans a trajectory around it in order to collect multiple views which are then registered together.

In the topic of object learning, Krause et al. [6] present an approach for one-shot learning of novel objects by interacting with a user through natural language. The approach relies on detectors and validators for the different attributes than can be used to describe objects. For instance, they use shape contexts to describe 2D shapes such as a red cross or a biohazard sign. This in turn requires models to be trained in advance and a prior knowledge of the kind of attributes that the user may refer to. Other work has approached object learning through developmental approaches, incorporating active exploration and social interaction [7], [8].

In a different line of work, the semantic web has been used to generate hypothesis over unknown objects based on their co-occurrence matrices with other identified objects in a scene. For instance, Young et al. [9] focus on the knowledge acquisition of task-related objects from the semantic web using contextual information obtained from situated observations. The approach assumes a partial knowledge of the scene in which the unknown object is found as well as a history of previous scenes where the object appeared. They make use of distributional semantics based on spatio-temporal context to propose label candidates, i.e., they compute the similarity among the identity of objects found in the scene and all possible label candidates in order to rank them. Later on, Young et al. [10] combined this approach with a pre-trained CNN. The difference is that in this case, they use the semantic web to refine the predictions output by the CNN. This however, suffers from the same shortcomings that pre-trained models usually have, i.e., there is a limited number of categories that they can predict. In both works, the system is not able to interact with users to improve its knowledge although the long-term goal is to present the curated list of candidates to a series of users (through crowdsourcing for example) for them to choose the correct one. In a somehow similar approach, Cartucho et al. [11] propose a learning algorithm that relies on predictions from a pre-trained network and user feedback to refine those predictions.

Vanzo et al. [12] proposed a model for incremental learning of objects but the focus is mainly on learning a dialogue policy for learning when to ask for feedback to the user based on the confidence scores given by the vision module. Other work [13] uses feedback from users to combine predictions from pre-trained networks in order to identify novel objects.

The work most similar to ours was recently proposed by Hasler et al. [14]. They developed a similar architecture that learns objects incrementally by interacting with users. However, their architecture differs from ours in that they don't use depth information for the feature representation, the image background is not discarded and they restrict the language the user can use to teach the robot. More importantly, they store all features whereas we use an incremental learning model that automatically compresses the data by learning prototypical feature vectors.

III. PROPOSED SYSTEM

Fig. 2 illustrates the system architecture and how every component interacts with each other. In the following sections we describe each of these modules in more detail.

A. Dialogue Manager

The dialogue manager is based on Opendial [15] and acts as the central hub of the system. It routes data across modules and manages the actions that need to be executed.

Even though it is possible to handle the natural language understanding portion with Opendial, it is usually limited by the pre-specified dialogue domain. Hence, in order to deal with open requests from the user, we rely on a separate Natural Language Understanding (NLU) module which is explained in Sec. III-B.

For generating the system's responses on the other hand, we use predefined templates.

B. Natural Language Understanding Module

The speech recognition module, based on the Google Speech API, submits the transcript of the user utterance to the dialogue manager which redirects it to the NLU module. The latter is responsible for recognizing the user's intent and the entity that is referred to in the utterance if relevant. The system can also make clarification requests and handle simple feedback given by the user.

1) Intent Recognition: For intent recognition, the Rasa[1] NLU API is used. This API has different pipelines for intent recognition and selecting one is generally based on the intended application and the amount of data available for training the respective model. In this work, the spacy-sklearn pipeline is used, which makes use of spaCy for feature extraction and an SVM for classification. This pipeline is suitable for small applications for which there isn't a lot of data available.

The module distinguishes among 4 different intents, *Recognize*, *RequestLearn*, *Rejection* and *Confirmation*. The first two prompt the system to either recognize the object in front or begin the learning behavior. The last two are used to accept feedback when the system asks for clarification. For training the model, we manually developed a small dataset of sentences that aims to capture the different ways in which people express the aforementioned intents.

2) Open-Set Entity Extraction: The entity extractor is realized with the dependency parser from spaCy[2] and is only relevant for the *RequestLearn* intent. When the user interacts with the system, every noun phrase (NP) in the user's utterance is analyzed. In order to decide whether to keep or discard the noun phrase, we look at the head of its root. We are only interested in those noun phrases that follow the verb *"be"* or the preposition *"of"* and we discard determiners and pronouns. For example, in the sentence from Fig. 3, there are 3 noun phrases, *"a box"*, *"cookies"* and *"the kitchen table"*. Following the given rules, we begin by discarding *"the kitchen table"* while the other two NPs are preserved. The preposition *"of"* expresses a relationship between entities and so we use it to join the NPs. Finally, determiners are removed such that the end result in the example is *"box of cookies"*.

C. Learning and Recognition Module

The learning and recognition module is based on the system proposed by Part and Lemon [16], the main difference being its implementation. In this work, we use ResNet50 [17] models as the pre-trained networks (available from Keras[3]) and Grow When Required [18] (GWR) self-organizing networks. In addition, every class is represented by its own

[1]https://rasa.com/
[2]https://spacy.io/
[3]https://keras.io/

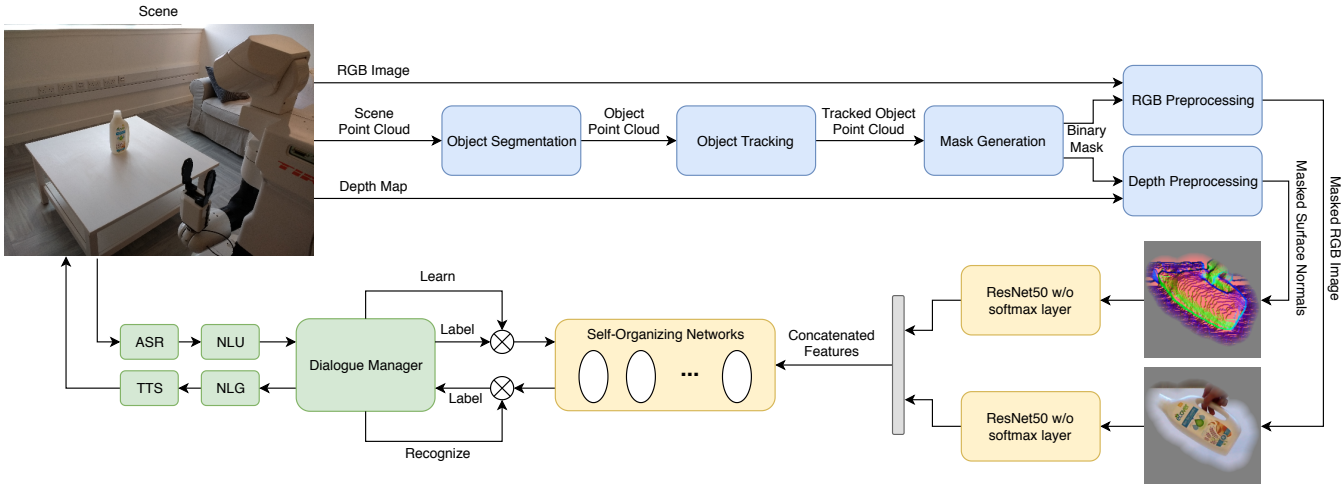

Fig. 2. System architecture. The scene point cloud is processed in order to segment and track the object of interest and eventually compute a binary mask that is used to process the RGB image and depth map (blue blocks). The learning model (yellow blocks) is then fed with the processed RGB and surface normals images in order to perform either learning or recognition. The dialogue system (green blocks) determines the user's intent and extracts the referred entity if applicable in order to manage the learning model's behavior (learning or recognition).

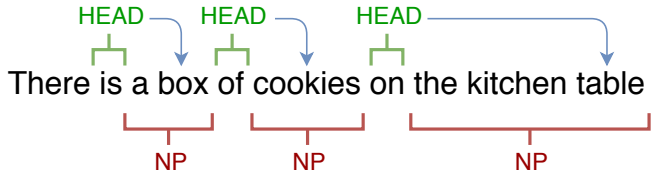

Fig. 3. Example of sentence processing for open-set entity extraction. NP are the noun phrases and the blue arrow points to the root of the noun phrase. Following the process described in Sec. III-B.2, the extracted entity for this example is *"box of cookies"*.

growing self-organizing network as opposed to having a single network for all the classes. This has the advantage that different classes won't interfere with each other and thus helps alleviate catastrophic forgetting, especially since the representations are fixed by the pre-trained networks.

The preprocessing pipeline is also similar to the one proposed by Part and Lemon [16] although in this case, we are dealing with raw data as opposed to a processed dataset and hence, the objects need to be segmented and tracked in order to produce suitable data for the learning module. Below, we describe each component in more detail.

1) Object Segmentation and Tracking: The pipeline begins by applying a pass-through filter and downsampling the point cloud, and looking for planar surfaces in the space in front of the robot. The plane is segmented using the RANdom SAmple Consensus (RANSAC) algorithm and its convex hull is computed on the result. The convex hull is used to focus the search for clusters only in the volume on top of the segmented plane. Finally, Euclidean clustering is applied to segment the object on the plane if any.

Once the object has been segmented, its point cloud is used to initialize a particle filter tracker. By doing so, we can track the pose of the object while it's being moved in front of the robot during the learning and recognition phases[4]. Tracking the object point cloud allows us to compute its binary mask by back-projecting the point cloud to the image frame using the camera parameters.

2) Image Preprocessing: Once the object's binary mask has been generated, it is used to find the bounding box containing the object, which is squared and then used to crop the images.

To be able to use the pre-trained ResNet50 model on depth information, the depth map has to be colorized first since the model was designed to process color information (RGB channels). The chosen colorization method is to compute surface normals on the depth map as described by Part and Lemon [16].

Finally, using the binary mask, the image background is gradually faded into the mean image of the original dataset used to train the ResNet50 models in order to reduce these models' response to background information.

3) Learning and Recognition Pipeline: The preprocessed color image and surface normals are fed to pre-trained ResNet50 models to extract their features. Both networks are exactly the same and were originally trained on ImageNet. In order to use them as feature extractors, the top layer was removed. Then, the features obtained from both channels are concatenated and fed to an array of incremental self-organizing networks.

The self-organizing networks are collections of nodes joined by edges that represent some data distribution. Every node has an associated weight and a firing counter that indicates how well trained the node is. If the firing counter is high, it means that the node has fired many times and as a result, it has learned the data it represents well and future firings will not have such a big impact on its weight. On the contrary, if the firing counter is low, that means

[4]Note that this also applies if the robot would be manipulating the object itself.

that the node is still learning and is susceptible to larger changes. The edges have an associated age that can increase or decrease depending on the stimuli produced by the data and are eliminated when the age reaches a certain maximum value.

During the learning behavior, the model searches for the network associated to the class corresponding to the label given by the user. If the associated network doesn't exist yet, the model creates and initializes a new one. If the network does already exist, then the model updates it with the new data. The update process is as follows. At the start, the network is initialized with the first two data points that become available, i.e., two nodes are inserted and their weights are set to the feature vectors of the incoming data points (images in this case). For every subsequent data point, we look for the two closest nodes in the network according to a similarity metric, e.g., Euclidean or cosine distances, and join them by an edge if they are not already joined. Then, we compute the firing activity of the best matching node according to (1):

$$a = e^{-||x-w_i||}, \tag{1}$$

where x is the feature vector of the input data point and w_i is the weight of the best matching node. If the activity is low and the node is well trained (has fired repeatedly in the past), then a new node is added between the two best matching nodes. On the other hand, if the activity is high or the node is relatively new (low firing counter), then the node and its neighbors are trained according to respective learning rates. In this manner, the network evolves to represent the underlying distribution of the data. To some extent, it can be thought of as a lossy data compression facility since it learns prototypes for the input data that then can be used to identify similar data.

During recognition, the Euclidean distance between the input feature vector and the weights of every node in every network is computed. The classification result is then selected as the label of the network that yielded the shortest distance among all networks.

IV. EXPERIMENTAL EVALUATION

A. Hardware Setup

For data collection, we used the Tiago robot developed by PAL Robotics and the RGB-D sensor mounted in its head, i.e., an Asus Xtion. Most of the processing however was performed on a development laptop connected to the robot through an Ethernet cable. The development laptop runs Ubuntu 16.04 LTS with ROS Kinetic and has an Intel Core i7 processor and an NVIDIA GeForce GTX 980M GPU. Speech was acquired with a wireless Singstar microphone and produced with Tiago's speakers.

B. Data Collection

The data was collected in a home environment[5] over the course of several days using 2 locations (the coffee table in the living room and the kitchen table) and varying illumination conditions, both natural and artificial. The setups used resemble the one shown in Fig. 1. Through natural language interaction, the user prompted the system to learn a series of objects and the data was acquired with the RGB-D sensor mounted on the Tiago robot, i.e., an Asus Xtion. We gathered 63 objects belonging to 26 classes. For every object and environmental setup, we collected 30 color images, depth maps and binary masks at a maximum rate of 5Hz. Variations in the acquisition rate are due to, among other things, the size of the object, i.e., the size of the segmented point cloud.

Fig. 4. The collection of common household objects used in this work. The dataset consists of 63 objects belonging to 26 classes.

Objects belonging to the same class vary either in appearance, shape or size. Our computed features do not take into account the size of the objects but this is often a useful piece of information since for certain cases it tends to be the distinguishing factor, e.g., "bring me the small cooking pot".

For collecting the data we took into consideration factors like location and illumination conditions. We also varied the side from where the demonstration was performed, i.e., where the user's hand appears in the robot's sensor feed. Other environmental factors included shadows, reflections, glares and backlight[6].

C. Training and Evaluation Protocol

1) Incremental Training and Evaluation: At the beginning, we trained the model on a subsampled[7] version of the Washington RGB-D Object dataset [19] to simulate the case in which the robot already has a base knowledge. This is particularly relevant for the experiment in Sec. V-C.

Our dataset is split into 4 different sessions. Each session is characterized by a change in location and illumination conditions. Fig. 5 shows examples of object images from the different sessions.

[5]A demo of the system is available at https://youtu.be/mWwwrW8eUfI.

[6]We omitted translucent objects in our study since depth information is too noisy in those cases.

[7]We took one every 25 frames.

(a) Session 1 (b) Session 2

(c) Session 3 (d) Session 4

Fig. 5. Examples of the environmental conditions for each session. a) Sunlight coming through the window in location 1. b) Natural light in location 1. c) Artificial light in location 2. d) Natural light in location 2.

Every experiment from Sec. V-B was run 10 times varying the random sequence in which the data was fed. For every run, we randomly select 3 sessions for incremental training and the remainder session only for evaluation. In addition, from every training session, we randomly select one instance from every class for evaluation, such that the final evaluation set consists of one entire session and a small subset of the other 3 sessions. If the class from the training session only has one instance, then it is used both for training and evaluation[8].

There are two conditions that we evaluate as discussed in Sec. V-B. First, the case in which the instances from different sessions don't mix during the learning process, i.e., the robot learns all the objects for one session before moving to the next. This is the sequential experiment. For the second case, we randomly shuffle instances from different sessions altogether.

2) Accuracy Evaluation when using Word Embeddings: For this experiment, we repeated a similar protocol as the one explained in the previous section for the sequential experiment. The main difference is that we used word embeddings to compute similarity measures between the labels predicted by the model and the ground-truth labels. We took into account 5 thresholds to report accuracy, i.e., 0.5, 0.6, 0.7, 0.8 and 0.9. The higher the threshold, the more selective the model will be. We run each experiment only once with the same random seed since in this case we only want to compare how the threshold affects the accuracy curves.

[8]Since we report **incremental** accuracy, this does not bias our results significantly. In addition, there is always at least one session that is fully used for evaluation such that there will be some degree of novelty in the evaluation set regardless of what stage the learning process is in.

V. RESULTS

A. Sample Dialogues

The following are some example interactions that the system is able to handle[9]. The first example illustrates the case in which the system gives a wrong recognition result:

User: What is this?
Robot: It's a shampoo.
U: No, it's laundry detergent.
R: Did you say laundry detergent?
U: Yes.
R: Ok, could you show me the laundry detergent?
Learning in progress (user moves object)...
R: I finished learning the laundry detergent.

The second example illustrates a case where the system extracts the wrong entity (or an incomplete entity[10]):

User: This is a soda can.
Robot: Did you say soda?
U: No.
R: Ohh, I see... Could you say just the name of the object again please?
U: Soda can.
R: Did you say soda can?
U: Yes.
R: Ok, could you show me the soda can?
Learning in progress (user moves object)...
R: I finished learning the soda can.

B. Incremental Accuracy

Figs. 6 and 7 show the incremental accuracy of the system on the proposed dataset for two different conditions. The first one assumes structured study sessions where the robot learns the objects in one location before moving to the next one. This scenario is useful to analyze how environmental factors affect system performance since that is the main difference among sessions. It is clear from Fig. 6 that the system performs relatively well after just one study session, reaching almost 60% accuracy. Note as well the sudden slight drops in performance with every session switch. We attribute this to the fact that during these switches, the model is fed with new data that follows a different distribution than the data learned in the previous sessions (due to changes in the environmental conditions) and a higher rate of misclassification occurs as a result. After more data from the new session continues to be incorporated, the system recovers.

In the second condition, the instances from all the sessions are mixed and learned in random order. Thus, the system gets accustomed to the different environmental conditions from the beginning of the learning process. As a result, the curve in Fig. 7 is smoother and has less variation across different runs. Note that this unstructured scenario is more realistic for

[9]The robot notifies the user that it finished learning once it has collected 30 sets of images as explained in Sec. IV-B

[10]In this case, this happens because the model spaCy relies on tags *"can"* as a verb instead of a noun and *"soda"* as the noun instead of a noun modifier. Hence, the noun phrase is just *"soda"*. Note however that the user may be ok with this result.

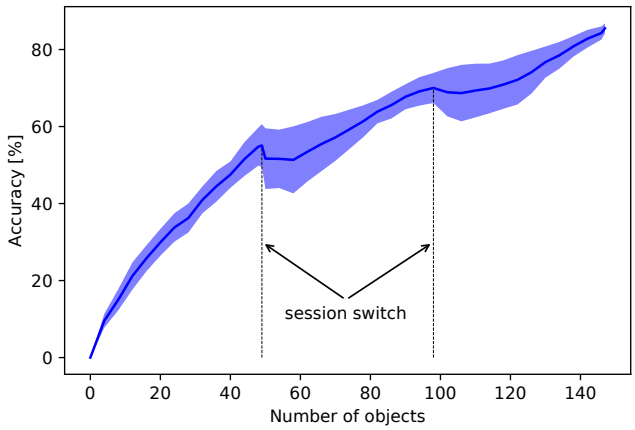

Fig. 6. Incremental accuracy computed over 10 runs on the proposed dataset when structuring the learning through well defined study sessions. The incremental evaluation is performed every 4 instances (objects) on the full evaluation set.

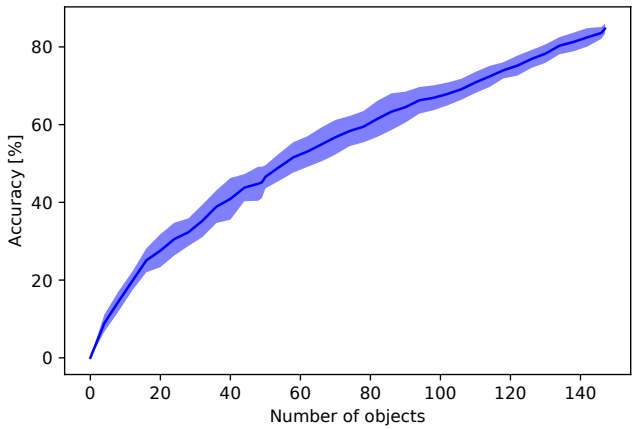

Fig. 7. Incremental accuracy computed over 10 runs on the proposed dataset when shuffling the instances across the different learning sessions. The incremental evaluation is performed every 4 instances (objects) on the full evaluation set. This evaluation condition corresponds to a more natural interaction since there is not a lot of structure in the learning, it happens as it is required.

a robot operating in the real world since there, there are no timetabled study sessions; the robot needs to adapt quickly and seamlessly when required.

One more interesting aspect is that after learning the full set of objects, the accuracy on the Washington RGB-D Object Dataset (base knowledge - see Sec. IV-C.1) was not affected, only a 1% drop was observed.

C. Effects of Incorporating Word Vector Similarity

Languages are very rich and people often use a wide variety of ways to refer to the same things, e.g., cup and mug, laptop and notebook, pitcher and jug. This poses some challenges when a robot has to interact with different users if it's not fit with a mechanism to disambiguate and unify its knowledge. One possible way to address this issue is to use word embeddings and compute similarities between

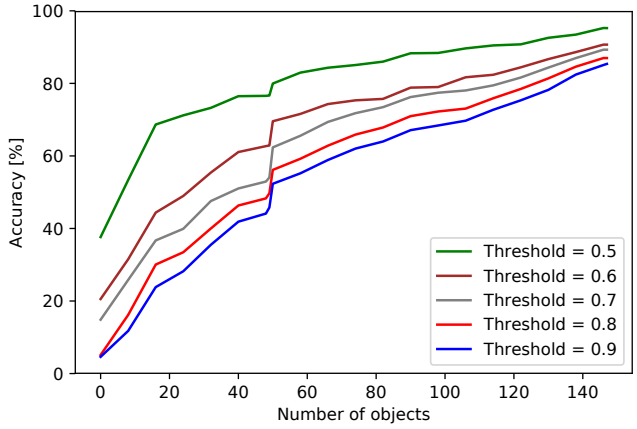

Fig. 8. Influence of using word vector similarity between the ground-truth and the predicted labels during the recognition phase. The incremental evaluation was performed every 8 instances. A lower threshold yields a higher accuracy as expected but it also means accepting less similar predictions as correct. In fact, thresholds below 0.7 are too permissive and may yield a decrease in perceived performance.

referring expressions[11]. For example, if the user requests the robot to "bring the jug", but the robot is not familiar with that label, it could look through its database and compute similarities to all known labels. Finally, it would ask the user if they mean "the pitcher", and update its database accordingly. Alternatively, it could bypass the clarification step and bring "the pitcher" if it is sufficiently confident that "jug" and "pitcher" refer to the same object.

Fig. 8 shows the classification accuracy when the system is allowed to tag a prediction as correct if it is sufficiently similar to the ground-truth according to an embedding model, the ground-truth assumed to be the requested item by the user. A higher similarity threshold corresponds to a more selective model where the ground-truth and retrieved label need to be very similar in order for the system to accept them as referring to the same object. Conversely, a lower similarity threshold will allow the system to accept less similar labels as good candidates. As illustrated in the figure, this will yield higher accuracy but at the expense of making more errors in the eye of the user, who may disagree about a "plier" being the same as a "screwdriver"[12]. Hence, we believe this mechanism to be very useful for allowing the robot to prompt the user for clarification and feedback but not as a way to bypass knowledge gaps or make confident assumptions.

VI. DISCUSSION AND FUTURE WORK

In this work, the data has been collected by a single user which does not allow us to make claims of generalization of the proposed architecture to different users. Part of our future work includes setting up experiments with non-expert users to address this limitation. The goal is to perform

[11] Another possibility could be to use an ontology like WordNet or any ontology that encodes taxonomic relationships.

[12] According to the model used, the similarity score between "plier" and "screwdriver" is about 0.75, which makes sense since both are tools, albeit with very different functions.

both quantitative and qualitative evaluations in regards to the quality of the collected data, the robustness of the dialogue system and the subjective perception of how the system performs from the user's perspective. In addition, these experiments will give insights into limitations of the NLU module and expose any potential expressions not captured by the current dataset.

Another avenue for future work involves the related problem of open-set object learning and recognition. So far, the system doesn't have a mechanism to identify unknown objects. Thus, it will always assume that the "new" objects belong to one of the categories that were learned before (closed-world assumption) unless told otherwise. This limits the level of autonomy that it can have and it increases the burden on the user who has to check whether the system is able to correctly identify every object.

One of the main limitations of the current version of our system is that whereas it is able to recognize a small amount of intents, it cannot deal well with open dialogue. We explored the use of confidence scores reported by the intent recognizer but we found that there is no reliable value that can distinguish between known and unknown intents. This is a very challenging problem and a hot research topic.

Finally, there are also limitations that arise from the requirement to synchronize data coming from different components, e.g., the binary mask obtained from processing the point cloud has to be in sync with the image feeds. We found that sometimes this can affect the speed in which the data can be processed and thus, the acquisition can take longer. We expect that as hardware becomes more power efficient, robots will be fit with more powerful processing components that will allow us to overcome most of these issues.

VII. CONCLUSION

In this paper, we presented a first version of a robot architecture for lifelong object learning through situated interactions with a user. The system is composed of a dialogue manager, a natural language understanding module and a learning and recognition module. Through the natural language understanding module, the system is able to extract open-set entities and resolve misunderstandings through the request for clarifications. The architecture includes a preprocessing pipeline for segmenting and tracking objects in real-time, which allows the system to focus on the object of interest. Our preliminary evaluation showed that the system was able to learn effectively and quickly from very little data.

ACKNOWLEDGMENT

We would like to thank Mauro Dragone for granting us access to the Assisted Living Lab at Heriot-Watt University and to the Tiago Robot. We also thank the anonymous reviewers for their comments.

During the development of this work, Jose L. Part was funded by a James Watt scholarship from the School of Mathematical and Computer Sciences at Heriot-Watt University. The evaluation experiments were performed on the Robotarium Cluster supported by the Robotarium Grant (EPSRC Grant No. EP/J015040/1).

REFERENCES

[1] Z. Chen and B. Liu, *Lifelong Machine Learning*, 2nd ed., R. J. Brachman and P. Stone, Eds. Morgan & Claypool Publishers, 2018, vol. 38.

[2] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual Lifelong Learning with Neural Networks: A Review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[3] A. Collet, B. Xiong, C. Gurau, M. Hebert, and S. S. Srinivasa, "Herb-Disc: Towards Lifelong Robotic Object Discovery," *The International Journal of Robotics Research (IJRR)*, vol. 34, no. 1, pp. 3–25, 2015.

[4] R. Finman, T. Whelan, M. Kaess, and J. J. Leonard, "Toward Lifelong Object Segmentation from Change Detection in Dense RGB-D Maps," in *Proceedings of the European Conference on Mobile Robots (ECMR)*, September 2013, pp. 178–185.

[5] T. Fäulhammer, R. Ambruş, C. Burbridge, M. Zillich, J. Folkesson, N. Hawes, P. Jensfelt, and M. Vincze, "Autonomous Learning of Object Models on a Mobile Robot," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 26–33, 2017.

[6] E. Krause, M. Zillich, T. Williams, and M. Scheutz, "Learning to Recognize Novel Objects in One Shot through Human-Robot Interactions in Natural Language Dialogues," in *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, 2014, pp. 2796–2802.

[7] S. Ivaldi, S. M. Nguyen, N. Lyubova, A. Droniou, V. Padois, D. Filliat, P.-Y. Oudeyer, and O. Sigaud, "Object Learning Through Active Exploration," *IEEE Transactions on Autonomous Mental Development*, vol. 6, no. 1, pp. 56–72, 2014.

[8] N. Lyubova, S. Ivaldi, and D. Filliat, "From Passive to Interactive Object Learning and Recognition through Self-Identification on a Humanoid Robot," *Autonomous Robots*, vol. 40, no. 1, pp. 33–57, 2016.

[9] J. Young, V. Basile, L. Kunze, E. Cabrio, and N. Hawes, "Towards Lifelong Object Learning by Integrating Situated Robot Perception and Semantic Web Mining," in *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 2016, pp. 1458–1466.

[10] J. Young, L. Kunze, V. Basile, E. Cabrio, N. Hawes, and B. Caputo, "Semantic Web-Mining and Deep Vision for Lifelong Object Discovery," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2774–2779.

[11] J. Cartucho, R. Ventura, and M. Veloso, "Robust Object Recognition Through Symbiotic Deep Learning In Mobile Robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2018.

[12] A. Vanzo, J. L. Part, Y. Yu, D. Nardi, and O. Lemon, "Incrementally Learning Semantic Attributes through Dialogue Interaction," in *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Stockholm, Sweden, July 2018, pp. 865–873.

[13] M. de Jong, K. Zhang, A. M. Roth, T. Rhodes, R. Schmucker, C. Zhou, S. Ferreira, J. Cartucho, and M. Veloso, "Towards a Robust Interactive and Learning Social Robot," in *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Stockholm, Sweden, July 2018, pp. 883–891.

[14] S. Hasler, J. Kreger, and U. Bauer-Wersing, "Interactive Incremental Online Learning of Objects Onboard of a Cooperative Autonomous Mobile Robot," in *International Conference on Neural Information Processing (ICONIP)*. Springer International Publishing, 2018, vol. LNCS 11307, pp. 279–290.

[15] P. Lison, "Structured Probabilistic Modelling for Dialogue Management," Ph.D. dissertation, University of Oslo, 2014.

[16] J. L. Part and O. Lemon, "Incremental Online Learning of Objects for Robots Operating in Real Environments," in *Proceedings of the 7th Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EPIROB)*, September 2017.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[18] S. Marsland, J. Shapiro, and U. Nehmzow, "A Self-Organising Network that Grows when Required," *Neural Networks*, vol. 15, no. 8-9, pp. 1041–1058, October 2002.

[19] K. Lai, L. Bo, X. Ren, and D. Fox, "A Large-Scale Hierarchical Multi-View RGB-D Object Dataset," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 1817–1824.