Incremental On-Line Learning of Object Classes using a Combination of Self-Organizing Incremental Neural Networks and Deep Convolutional Neural Networks*

Jose L. Part[1] and Oliver Lemon[2]

***Abstract*— This paper reports initial results from our ongoing work on Natural Language grounding through situated interaction for robots acting in unstructured environments. We present an approach for incremental on-line learning of object classes based on a novel combination of a Self-Organizing Incremental Neural Network (SOINN) and a deep Convolutional Neural Network (CNN). We evaluate our approach on the RGB-D Object Dataset and on four aspects: 1) object classification accuracy, 2) incremental learning in terms of the number of classes, 3) potential for realising active learning, and 4) feasibility to use the system in an on-line setting. Our results show that the approach has an acceptable performance for object classification (e.g. over 90% accuracy) and provide some insights into how it could be improved by harnessing interaction with both the environment and a human tutor.**

I. INTRODUCTION

In order to take robots out of the lab and deploy them in the real world, they need to be able to understand the environment in which they operate and to communicate effectively with people. It is not enough to have them execute pre-programmed routines and deal only with objects and in situations they have been trained on. They have to be able to rapidly learn and adapt to new environments and the people with whom they are going to be interacting.

One step towards this goal is to develop methods that allow robots to learn to recognize new objects as they interact with them, *i.e.* incrementally and in an on-line fashion.

In the last few years, the image classification scene has been dominated by approaches based on multiple stacked layers of Convolutional Neural Networks ([1], [2], [3]). These models are usually designed to solve a particular task and are trained off-line on massive datasets of hand-labelled images. Some of these approaches have been tailored for the object recognition task by using a combination of RGB and depth information ([4], [5]) and fine-tuning or otherwise using pre-trained CNNs ([6], [7], [8]). A further problem for such methods is that it is assumed that the number of classes is known in advance. Thus, once trained, if the task changes or new classes need to be added, the whole model has to be retrained. All these factors limit their suitability for real-world robot applications, where the data is not available in advance, the number of classes is not known, and the model needs to be trained on-line and with a limited number of examples.

Pasquale *et al.* [9] investigated how to leverage the success of deep learning architectures in an incremental setting by using the extracted features from a pre-trained Convolutional Neural Network (CNN) as input to a linear classifier that can be updated every time new samples become available. However, they address incrementality from the point of view of the amount of data rather than the number of classes. Thus, their approach does not support the incorporation of new classes to the model once it has been initially trained.

Nyga *et al.* [10] used an ensemble of experts coupled with a Markov Logic Network (MLN) to classify common household objects. The ensemble is composed by a series of algorithms that annotate individual features (evidence) of the observed object and the MLN infers the object's class based on that evidence. The main disadvantage of their approach is that the MLN is trained off-line using a dataset of hand-labelled scenes. Moreover, the ensemble algorithms are hand-engineered and fixed at training time. Hence, this approach is not suitable for incremental on-line learning.

Other methods for object recognition focus on zero- or one-shot learning ([11], [12], [13], [14]), which are essentially attribute-based classification approaches. However, they still require to have pre-trained classifiers for each attribute that is being considered. Hence, they suffer from the same shortcomings mentioned before.

We argue that in real-world applications we cannot foresee all the objects that the robot will interact with and hence, propose to use an incremental on-line learning approach. In order to state our problem more clearly, we make the following assumptions:

1) The data is not available beforehand, which means that the number of classes is unknown and labels, if any, are provided at runtime.
2) The data is input sequentially (*e.g.* video frames), which means that there is a high correlation and temporal coherence between consecutive data points.
3) There is a human tutor available that is able to provide labels at runtime.

In this work, we aim to assess how well our approach discriminates between different classes, how it behaves when adding new classes incrementally, how suitable it is for

*The lead author is being supported by a James-Watt scholarship from the School of Mathematical and Computer Sciences at Heriot-Watt University. This work is partially being supported by the ROBOTARIUM Grant (EPSRC Grant No. EP/J015040/1) and by the European Union's Horizon 2020 programme under grant agreement No. 688147 (MuMMER project http://mummer-project.eu)

[1]Jose L. Part is with the HRI Lab at Heriot-Watt University and with the Edinburgh Centre for Robotics, Scotland, UK jose.part@ed.ac.uk

[2]Oliver Lemon is with the Interaction Lab and the HRI Lab at Heriot-Watt University, Scotland, UK o.lemon@hw.ac.uk

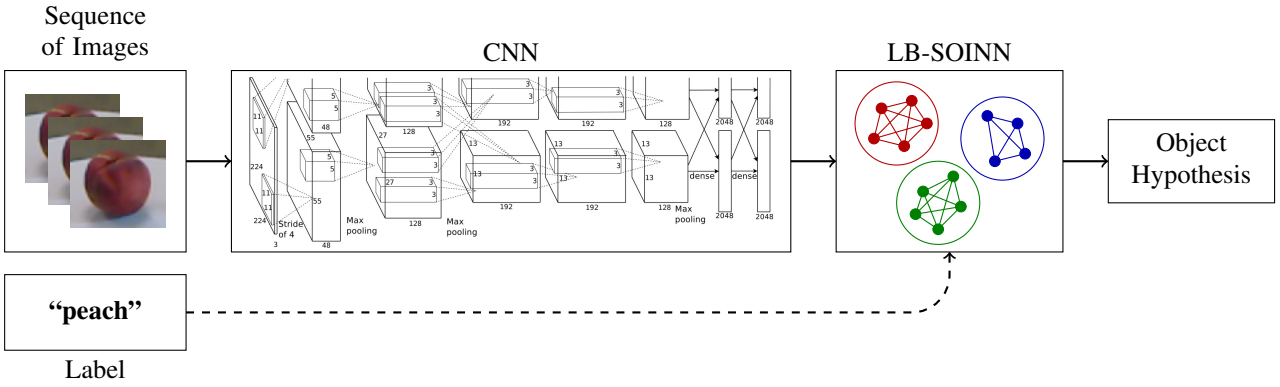


Fig. 1. Simplified diagram of the system architecture. Images of the target objects are input sequentially to the CNN, based on [1], which outputs the computed feature vectors. These feature vectors are then used to train a Load-Balancing SOINN classifier that learns the topology of the data and clusters it according to a similarity metric. Currently, most of our evaluation has been performed using the approach as a supervised method (dashed line).

discovering knowledge gaps and acting appropriately to address them (active learning), and how feasible it is to use it as an on-line learning method.

The remainder of the paper is organized as follows: Section II presents our proposed method and the rationale behind it, Section III describes the dataset we used, Section IV explains the experiments we performed and the results we obtained, and finally Section V discusses our results and future directions of work.

II. METHODS

In order to be in line with our motivation and assumptions, we propose the system architecture illustrated in Fig. 1, which consists of two main blocks, namely a feature extractor (CNN) and a classifier (LB-SOINN). Images are input sequentially into the feature extractor, which computes a 4096-dimensional feature vector, and this vector is then fed to the classifier, which learns the underlying structure of the data. We now describe each component in more detail.

A. Classifier

In order to deal with the particular nature of our problem, we adopted the Load-Balancing version of the Self-Organizing Incremental Neural Network (LB-SOINN) proposed by Zhang *et al.* [15] as our classifier. LB-SOINN is an unsupervised learning method inspired by the Self-Organizing Map [16]. Each node in the network has an associated weight, which lives in the feature space of the data. Every time a new signal (feature vector) becomes available, the algorithm assesses whether a new node should be added to the network based on a similarity metric between the input signal and the weights of the two nearest nodes. If no new node is added, the weights and connections of the existing network get updated. In this manner, the topology of the network is continuously being updated to reflect the distribution of the input data.

Some of the claims [15] that motivated us to select this approach are that LB-SOINN: 1) is able to cluster irregular data distributions, 2) is able to process both stationary and non-stationary data, 3) can deal with overlapped regions, 4) is robust to noise in the input data, 5) preserves previously acquired knowledge, and 6) forms a stable structure. However, the method still has the disadvantage that it depends on six parameters that need to be tuned depending on the nature of the input data. These are λ, which determines the frequency of noise removal, age_{max}, which determines the lifetime of each edge in the network, c_1 and c_2, which control the deletion of nodes that are attributed to noise, γ, which is used for controlling how classes are split and merged, and η, which controls the ratio between different distance metrics for assessing the similarity between feature vectors. For computing the similarity between nodes, we adopted the framework proposed by Zhang *et al.* [15], which gives different weights to the Euclidean distance and the Cosine distance depending on the dimensionality of the input data.

Even though the approach belongs to the unsupervised learning category, we have used it as a supervised method. This is for two main reasons. First, we assume that this method will be used in an interactive setting were there is some form of supervision available. Second, depending on how the data is distributed, there is the possibility that multiple clusters form for the same class (see Section IV-A for more details), in which case a purely unsupervised method would lead to having some classes split into multiple subclasses. However, the fact that we are using an unsupervised method allows us to enable a behaviour of *knowledge gap detection* as will be discussed in Section IV-C. This not only has the potential to improve the representations learnt by the robot but also to reduce the load for the tutor in an interactive setting.

Another difference between the approach proposed by Zhang *et al.* [15] and our implementation is that we have disabled the algorithm for splitting overlapped regions since, at least for our data, it behaves quite unstably.

B. Feature Extractor

Another important part of our approach is the chosen feature representations. Despite all the limitations that deep learning architectures have, as discussed in Section I, it has been shown ([17], [18], [19]) that deep Convolutional

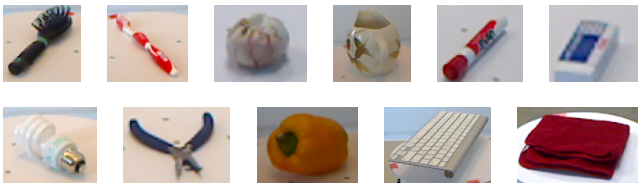


Fig. 2. Sample images from the RGB-D Object Dataset.

Neural Networks (CNNs) that have been trained on massive datasets can be used as off-the-shelf feature extractors for a variety of visual tasks obtaining state-of-the-art results. These architectures learn hierarchical representations of the input data that can be divided into low-level at the beginning of the network and high-level (semantic information) at the end. This also has the advantage that we no longer need to resort to hand-engineered features that require time and expertise to be developed.

In accordance with the previous arguments, we chose the well-established AlexNet model based on the network proposed by Krizhevsky *et al.* [1] and publicly available in the *Caffe Model Zoo* [20]. This network was trained on a subset of the *ImageNet* dataset and is composed of five convolutional layers and three fully-connected layers. The last fully-connected layer is a softmax layer that outputs a probability distribution over all the classes. Since we are not interested in the classes for which the network was originally trained, we discarded the last layer. After running tests using the feature vectors obtained after the last convolutional layer and the two remaining fully-connected layers, we decided to use the features obtained after the last fully connected layer because these led to better classification results.

III. DATASET

In order to evaluate our approach, we selected the RGB-D Object Dataset [21], which is composed of 300 common household objects organised into 51 categories. The dataset was acquired with a Kinect v1 sensor by placing each object on a turntable and taking video sequences of a full rotation at three different heights of the sensor. For every frame, there is RGB and depth information available, though for our preliminary study we have only used the RGB information. Samples of the sort of images included in the dataset can be seen in Fig. 2.

The main reason that we chose this dataset for the evaluation of our approach is that it provides different views of the same object in some sort of sequence. This is of particular interest because it resembles to some extent how a robot could interact with the objects in order to get more information and build a better, more complete representation. Thus, we suggest that the results obtained from our evaluation can give valuable insights into how our approach would perform on a real robot in a real environment.

IV. EXPERIMENTS

Below we propose a series of experiments that aim to evaluate our approach in terms of classification accuracy, robustness against addition of new classes (incrementality), and suitability for realising active and on-line learning.

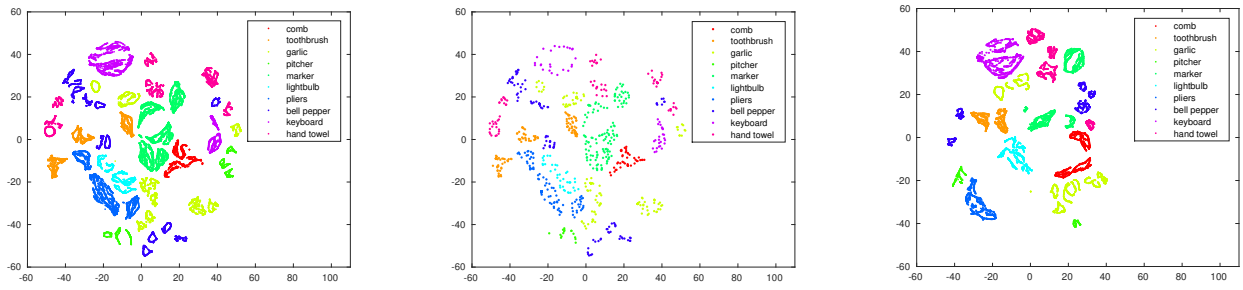
A. Object Classification

For evaluating the classification accuracy of our approach, we selected 10 random classes from the dataset and divided them into 60% of the instances for training and the remaining 40% for testing. The parameters of the LB-SOINN were chosen as $\lambda = 100$, $age_{max} = 50$, $\eta = 1.001$, $c_1 = 10.0$, and $c_2 = 0.1$. These values were obtained empirically through a non-exhaustive search in the parameter space of λ, age_{max}, c_1, and c_2. For η we used the value proposed by Zhang *et al.*, which reduces the weight of the Euclidean distance for the similarity calculation as the dimensionality of the input data increases. Since we disabled the algorithm for splitting overlapped regions, the parameter γ was not used.

We observed that when using the method in a purely unsupervised manner, more clusters than the alleged number of classes were being reported by the system. We believe that this is a consequence of how the data is distributed in the feature space. Presumably, for some classes we may have a higher intra-class variation that results in the formation of several clusters that belong to the same superclass. In order to investigate this further, we decided to use a variant of the t-distributed Stochastic Neighbour Embedding (t-SNE) algorithm [22] that uses Barnes-Hut (BH) optimization for projecting our data onto a 2-dimensional space. Despite the fact that it is difficult to draw conclusions from these projections, they can be quite useful to get some insights into how the data is distributed and what the model is learning. Fig. 3 shows the projections of the training data, the network (nodes) learnt by our approach, and the test data respectively, for one of the runs. From this projection, we can sense that effectively there is more than one cluster for most of the classes. Moreover, it seems that some classes are more concentrated whereas others are spread across the whole feature space. These insights coupled with the assumption that there is a tutor available are what motivated us to use the method in a supervised manner.

Since this dataset was built in a controlled environment (*e.g.* the objects are rotated in front of the sensor in a systematic way), we decided to randomly shuffle the images per instance in order to emulate a more natural interaction (this is consistent with moving the object randomly in front of the robot).

Fig. 4 shows the results of training and testing the algorithm on 10 classes for 20 different runs, where each run is characterised by a different random sequence of the classes. The box plots show how the accuracy per class varies in every execution and are an indication of the dependence that exists on the sequence in which the system learns the objects. As can be seen, the accuracy is consistently high (over 90%) for most of the classes. In the case of the "bell pepper", for which the accuracy is relatively low, we observed that this was an effect of how the dataset was split. In particular, for this class the system was being trained on red and green peppers, and tested on green and yellow



(a) (b) (c)

Fig. 3. Projections of the a) training set, b) learnt network, and c) test set. The data is first projected onto a 50-dimensional space using Principal Component Analysis (PCA) and then onto a 2-dimensional space using the t-SNE algorithm with BH optimization. The labels for the learnt network were assigned through supervision.

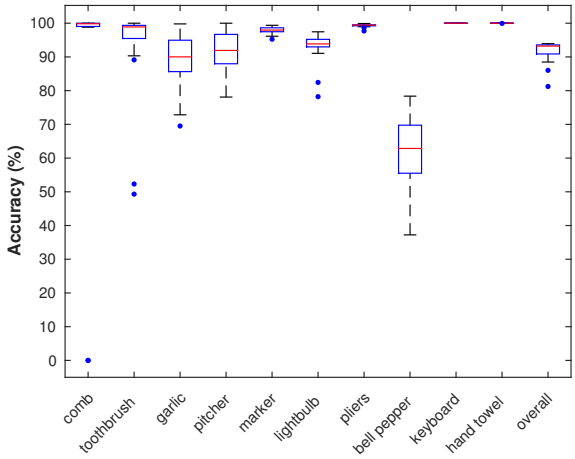


Fig. 4. Box plots of the accuracies for 10 classes (54 instances). The box plots have a maximum whisker length of 1.5 IQR and were generated by running the algorithm 20 times with the classes being input in random order in each run. 60% of instances were used for training and 40% for testing. The last box plot corresponds to the overall accuracy for the 10 classes.

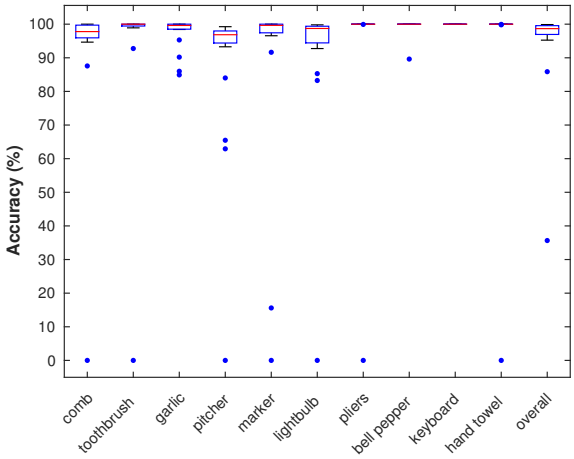


Fig. 5. Box plots of the accuracies for 10 classes (54 instances). The box plots have a maximum whisker length of 1.5 IQR and were generated by running the algorithm 20 times with the classes being input in random order in each run. All instances were used for training and testing. 60% of images were used for training and 40% for testing. The last box plot corresponds to the overall accuracy for the 10 classes.

peppers. As discussed previously, in Fig. 3 we saw that for some classes we had multiple clusters. In most of these cases, these subclasses correspond to different instances of the same class (high intra-class variation). Consequently, the system can get confused when it sees an instance that is quite different from the ones it saw during training. In order to test this hypothesis, we repeated the previous experiment but we split the dataset per number of images rather than per number of instances. The corresponding box plots are shown in Fig. 5, where we can see that the accuracy is superior to 95% for all the classes.

In Fig. 5 we can also observe outliers with 0% accuracy for most of the classes. This issue requires further investigation but we believe that it is a consequence of particular sequences of the input data. It is possible that in some conditions only small isolated clusters will form, which will get removed periodically by the noise removal algorithm preventing the formation of robust structures.

B. Incremental Learning

In order to test how well our model behaves when incrementally adding new classes, we performed the following evaluation. Using the same partition of the dataset as before (60% - 40% instances), we began training and testing the system on only two classes and computing the overall accuracy. Then, we trained on two more classes and computed the overall accuracy based on the four classes. We repeated this process until having trained the system on 50 classes. Fig. 6 reports the results (blue line). As can be seen, the approach has a very good incremental performance for a relatively small number of classes but it decreases significantly above 18 classes. This has to do again with the fact that we did not use all the instances for training. In fact, by performing an analogous experiment to the one discussed in Section IV-A, we observed that when using all the instances for training, the overall accuracy remains above 90% (red line).

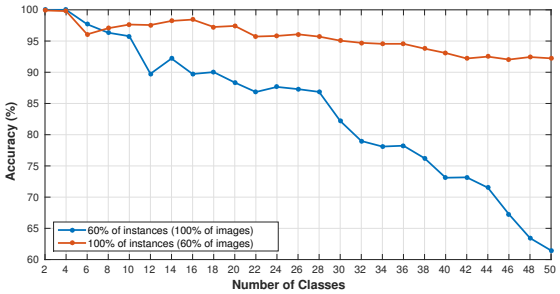


Fig. 6. Incremental learning. Evaluation of how the overall accuracy of the approach varies with the addition of new classes. We performed the evaluation with two partitions of the dataset as discussed in Section IV-A.

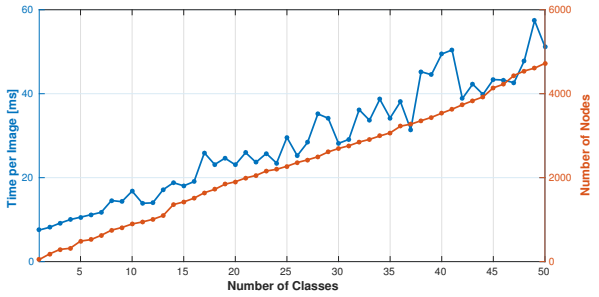


Fig. 7. On-line learning. Variation in the processing time per image and in the number of nodes as a function of the number of learnt classes.

C. Knowledge Gap Detection

In order to assess whether the model has the potential to identify knowledge gaps (*i.e.* detect when a new label is needed), which is necessary for realising active learning, we defined three different heuristics:

- Agreement between 2 nearest neighbours (NNA): A new data point belongs to a class if and only if the two nearest neighbours have the same label.
- Nearest neighbour threshold (NNT): A new data point belongs to a class if and only if the distance to its nearest neighbour is less than the similarity threshold.
- Confidence score (NNC): A new data point belongs to a class if and only if the ratio between similarity threshold and distance to its nearest neighbour is greater than a pre-defined value (confidence threshold).

The similarity threshold of a node in the network is a quantity that determines whether a new node should be added in the network [15] when a new signal is present. Hence, it is logical to assume that this quantity can be used for aiding in the classification of new data points and eventually deciding whether new nodes should be added in order to cover the corresponding area of the feature space. An analogous argument can be made about considering the agreement between the two nearest nodes, since this would mean that there is a higher probability of the node belonging to that class. If the corresponding heuristic is not satisfied, the system can identify the data point as unknown and switch to a learning behaviour.

We performed the evaluation on a small subset of the dataset with disjoint classes, *i.e.* we train on 5 classes and test on 5 different classes. The goal is to observe whether the model can identify what it does not know using any of the proposed heuristics. Table I summarizes the results. It is evident that whereas the NNT heuristic allows for a better discrimination of unknown classes, it also punishes harder on known classes as compared to NNA. On the other hand, NNC offers a better trade-off at identifying knowledge gaps without misclassifying known classes as unknown.

D. Computational Performance

One of the main goals of our research is to be able to perform the training and recognition processes in real time (on-line learning). To this end, we evaluated the time it takes for the system to process each image frame with respect to the number of learnt classes. We performed this evaluation in the incremental setting described previously. The results are summarized in Fig. 7. Despite the fact that the time increases almost linearly with the size of the network, it can be seen that the computational cost to process every frame is still suitable for on-line operation.

For all the tests, we run the feature extractor on a GPU NVIDIA GeForce GTX 980M and the classifier on a CPU Intel Core i7 with 32 GB of RAM. The operating system used is Linux Ubuntu 14.04 LTS 64-bit.

V. DISCUSSION AND FUTURE WORK

In this paper we presented a new approach for on-line incremental learning that builds on the successes of deep Convolutional Neural Networks while allowing for training the system in real time. We evaluated the approach in terms of classification accuracy, incrementality, and suitability for performing active and on-line learning. Our results show that the approach has a very impressive classification accuracy when training the system on all the instances available for each class, but it tends to decrease when evaluating the system on unseen instances of known classes. This shows that for some classes (depending on the level of intra-class variation) the approach is able to discriminate among different instances. Further investigations are required for assessing how to leverage this property while also retaining generalization over the classes.

We also showed that the approach is suitable for on-line learning although it is expected that with the number of classes and consequent increase in the number of nodes, the computation time will increase. However, most of the variation in computation time is due to traversing the network for finding the nearest neighbours, a task that is highly parallelizable. Hence, we do not consider this to be an issue.

Some of the negative features that we observed include the fact that the approach is highly dependent on the type of data, the sequence in which it is input to the algorithm, and the chosen parameters. Moreover, we saw that with the proposed heuristics it is difficult but still possible to discriminate unknown classes while still preserving a good accuracy on the classes that are known.

TABLE I

CONFUSION MATRICES RESULTING FROM THE USE OF THE HEURISTICS PROPOSED IN SECTION IV-C. CONFIDENCE THRESHOLD FOR NNC IS 0.5.

	NNA						**NNT**						**NNC**					
	Non-Disjoint Classes																	
comb	1176	0	0	0	1	12	331	0	0	0	0	858	944	0	0	0	6	239
toothb.	0	1147	0	0	0	21	0	188	0	0	2	978	0	874	0	0	2	292
garlic	0	0	2378	0	0	0	0	0	645	0	0	1733	0	0	1872	0	0	506
pitcher	0	0	0	557	0	0	0	0	0	0	0	557	0	0	0	351	0	206
marker	1	23	1	0	1845	67	0	0	0	0	629	1308	0	30	3	0	1782	122
	Disjoint Classes																	
lightb.	1	313	734	0	47	158	0	0	12	0	13	1228	0	267	539	0	82	365
pliers	668	0	80	0	116	175	9	0	0	0	0	1030	122	2	20	0	186	709
pepper	0	0	862	0	91	272	0	0	0	0	0	1225	0	20	98	0	257	850
keyb.	814	0	1	40	30	186	0	0	0	0	0	1071	181	0	0	7	10	873
towel	57	0	724	516	0	283	0	0	0	0	0	1580	46	0	22	551	14	947
	comb	**toothbrush**	**garlic**	**pitcher**	**marker**	**unknown**	**comb**	**toothbrush**	**garlic**	**pitcher**	**marker**	**unknown**	**comb**	**toothbrush**	**garlic**	**pitcher**	**marker**	**unknown**

A more thorough evaluation is required with a higher number of classes in order to assess the scalability of the approach. So far, the number of classes has been limited by the available datasets, but in real-world applications there is no limit on the number of objects a robot may have to learn.

In the future, we plan to implement the approach on a real robot and evaluate the benefits that situated interaction may convey. For example, having a tutor with whom the robot could interact would allow it to correct wrong labels and learn stronger and more reliable structures. In addition, we could leverage the knowledge that time is continuous and that objects preserve their class labels across time independently from the viewpoint. In particular, Pasquale *et al.* [9] showed that using a temporal window in order to produce the class label at test time significantly improves recognition accuracy. Finally, we also plan to incorporate depth information into the feature representations, which has been shown to outperform RGB information for classification [7], and study further the potential for realising active learning.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1106–1114.

[2] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," 2014.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[4] R. Socher, B. Huval, B. Bhat, C. D. Manning, and A. Y. Ng, "Convolutional-Recursive Deep Learning for 3D Object Classification," in *Advances in Neural Information Processing Systems 25*, 2012.

[5] D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, September 2015.

[6] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal Deep Learning for Robust RGB-D Object Recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015.

[7] L. Madai-Tahy, S. Otte, R. Hanten, and A. Zell, *Revisiting Deep Convolutional Neural Networks for RGB-D Based Object Recognition*. Springer International Publishing, 2016, pp. 29–37.

[8] M. Schwarz, H. Schulz, and S. Behnke, "RGB-D Object Recognition and Pose Estimation based on Pre-Trained Convolutional Neural Network Features," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 1329–1335.

[9] G. Pasquale, C. Ciliberto, F. Odone, L. Rosasco, and L. Natale, "Teaching iCub to recognize objects using deep Convolutional Neural Networks." in *JMLR Workshop and Conference Proceedings*, vol. 43, 2015, pp. 21–25.

[10] D. Nyga, F. Balint-Benczedi, and M. Beetz, "Pr2 looking at things - Ensemble learning for unstructured information processing with markov logic networks," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 3916–3923.

[11] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer." in *CVPR*. IEEE Computer Society, 2009, pp. 951–958.

[12] P. Kankuekul, A. Kawewong, S. Tangruamsub, and O. Hasegawa, "Online incremental attribute-based zero-shot learning." in *CVPR*. IEEE Computer Society, 2012, pp. 3657–3664.

[13] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-Based Classification for Zero-Shot Visual Object Categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 453–465, 2014.

[14] 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 Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[15] H. Zhang, X. Xiao, and O. Hasegawa, "A load-balancing self-organizing incremental neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1096–1105, 2014.

[16] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, pp. 1464–1480, 1990.

[17] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2014.

[18] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 647–655.

[19] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems 27*, 2014, pp. 3320–3328.

[20] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM International Conference on Multimedia*, 2014, pp. 675–678.

[21] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset." in *ICRA*. IEEE, 2011, pp. 1817–1824.

[22] L. Van Der Maaten, "Accelerating t-SNE Using Tree-based Algorithms," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3221–3245, January 2014.