Automatic creation of large scale object databases from Web resources: a case study in robot vision

Dario Molinari[*,1], Giulia Pasquale[*,2], Lorenzo Natale[2], and Barbara Caputo[2,3]

[1] Sapienza Università di Roma, Italy, Italy
[2] Istituto Italiano di Tecnologia, Italy
[3] Politecnico di Torino, Italy
barbara.caputo@polito.it
[*]equal contribution

This is a post-peer-review, pre-copyedit version of an article published in:

Molinari D., Pasquale G., Natale L., Caputo B. (2019) Automatic Creation of Large Scale Object Databases from Web Resources: A Case Study in Robot Vision. In: Ricci E., Rota Bulò S., Snoek C., Lanz O., Messelodi S., Sebe N. (eds) Image Analysis and Processing – ICIAP 2019. ICIAP 2019. Lecture Notes in Computer Science, vol 11752. Springer, Cham. https://doi.org/10.1007/978-3-030-30645-8_45

Abstract. A fundamental ingredient in the success of deep learning for computer and robot vision is the availability of very large-scale annotated databases. ImageNet, with its 1000 object classes and 1.2 million images, tends to be the dominant data collection for creating pre-trained deep architectures. A less investigated avenue is how the possibility to create task-specific data collections on demand, with limited or without manual effort, would affect the performance of convolutional architectures. This would be useful for all those cases where contextual information about the deployment of the deep net is available, and it would be particularly relevant for robot vision applications, where such knowledge is usually available. The goal of this work is to present a protocol for the automated creation of task specific datasets starting from a pre-defined list of object classes, exploiting the Web as a source of information in an automated fashion. Our pipeline consists of (a) an algorithm for automatic Web crawling that searches for "image class seeds", i.e., informative images of object classes of interest, (b) algorithms for figure-ground segmentation of the object of interest and pasting of the segmented item in contextual images close to where the agent is going to work, and (c) a tailored data augmentation routine for maximizing the informative content of the generated images. A thorough set of experiments on a public benchmark, as well as deployment to a robot platform, prove the value of the proposed approach.

Keywords: Object Categorization · Robot Vision · Web Vision.

1 Introduction

Robots need to have a visual understanding of their surroundings in order to have cognitive behaviors. From visually perceiving an object, to recognizing it, to understanding what it is, what its properties are and how it should be acted upon, all these are crucial components to have truly intelligent and autonomous systems. Since the seminal work of Krizhevsky et al [9], the overwhelming majority of state of the art approaches in computer and robot vision for object recognition are based on Convolutional Neural Networks (CNNs, [11]), which use end-to-end architectures achieving feature learning and classification at the same time. Compared to shallow learning approaches, where feature extraction and classification are two separate steps often laded with heuristics, CNNs offer several advantages: first, they have proved over countless benchmarks to be able to achieve much higher accuracies on basically any visual recognition problem; second, they offer a conceptual simplicity of use that has made them very quickly the dominant learning tool of the community. Despite these advantages, they also present some limitations, such as high computational cost, long training time and the demand for large datasets, to name a few. As CNNs are data-hungry algorithms, the possibility to train a given model on very large scale annotated data collections is crucial for their success. As a consequence, architectures trained over ImageNet [2] are the cornerstone of the vast majority of CNN-based object recognition methods; such architectures are then adapted to various classification needs through fine-tuning. This again, in turn, requires annotated data collection and non trivial manual effort, although not of the same scale needed for end-to-end training of CNNs.

This paper addresses this issue, following the recent trend of developing algorithms for the automatic creation of annotated data from the Web through smart downloading approaches [13]. As opposed to dealing with the automatic creation of a very large scale data collection, that inevitably brings with it issues related to the percentage of noisy images downloaded and of their effect on the training of the network, we propose a protocol for generating automatically task-specific databases for the fine tuning of pre-trained architectures. Given a list of object categories that the robot is expected to encounter while performing its assigned task, we first search the Web for a limited number of images representing the object of interest, in white/empty backgrounds. By taking only the first images resulting from the search, we strongly limit the amount of wrong/noisy images in our download. Once obtained the images, we figure ground segment them to remove any possible artifact in the background, and we paste them on generic backgrounds resembling the environment where the robot will be deployed. Further data augmentation contributes to bridge the perceptual gap between images found on the Web and images that might be acquired in the actual robot setting. Figure 1 gives an overview of the overall protocol. We evaluated the contribution of each step of the data generation pipeline by fine-tuning a deep network to address an object categorization task on a publicly available benchmark (Figure 1, bottom left). We then deployed the pipeline on a robot platform (Figure 1,

bottom right), where we show that it can run "on-the-fly" to generate image sets for training standard SVM classifiers for fast object categorization learning.

The rest of the paper is organized as follows: after a review of relevant previous work (section 2), we describe the protocol proposed for the automatic creation of databases (section 3); section 4 describes the experiments performed and our findings, while conclusions and future works are discussed in section 5.

Fig. 1: Visual representation of the proposed system.

2 Related Works

Earlier work explored the possibility of mining the Web for semantic information to be used in robot systems, mostly to populate automatically on-board knowledge base representations [17, 19, 18]. Still, semantic information alone will not suffice: visual perceptual capabilities are crucial for robots to operate in unconstrained, task-oriented settings. As the leading deep learning paradigm for robot

vision relies heavily on the availability of data collections, the ability to recognize large classes of objects is linked to the creation of such data corpora.

Database creation from the Web has been attempted in the past with the use of semantic query expansions [1, 13], where the query expansion helps in reducing the amount of noisy and mislabeled images automatically downloaded, while at the same time helps in guaranteeing the visual richness of the collection. In spite of this, automatic data creation from the Web tends to include a non-trivial percentage of noise in the data, that might negatively affect the performance of convolutional networks trained on them. Several authors proposed strategies to deal with it [3]. Researchers have also started working on automatic data mining for robot vision applications with deep networks and the results are promising [13]. All the works revised above target explicitly the creation of general purpose databases, mimicking ImageNet.

We are not aware of previous work attempting to create task specific databases from the Web without manual annotation, nor attempting to use Web data as starting point for the creation of artificial, synthetic images.

3 Method

This section details the steps of our pipeline, as outlined in Figure 1.

Images Download From the Web. This step requires dealing with noisy images (i.e., images found when searching the Web for a given object that instead show something else). We address this issue by noticing that, for the vast majority of publicly available search engines, the top-retrieved images tend to depict the object of interest on an uniform background. Thus, we developed a simple script that, given a label, downloads the first N images found in the Web (after duplicate removal), that we use as "seed images", to be augmented with synthetic transformations. While this does not guarantee the lack of noise in the seeds, we verified heuristically that, by keeping $N \sim 100$, its impact is largely reduced.

Object Mask Extraction. For creating the synthetic images, it is first necessary to "extract" the objects of interest from the downloaded image. To do this automatically, we applied the foreground/background segmentation method from [7]. The authors showed that a fully convolutional network with backbone weights pre-trained on a large-scale image classification dataset (like ImageNet [2]) can learn to produce dense binary segmentation masks by fine-tuning on a relatively small set of images with foreground/background pixel-level annotations. The idea is to leverage the notion of "pixel objectness" learned by the network on the large classification task, and fine-tune it to 'extend' its activation responses from fragments to entire objects. We used the model released by [7], which worked well off-the-shelf. We note that, being the pipeline fully automatic, it has to deal with the noise in the masks.

Synthetic Data Generation. At this stage, the segmented "seed" objects must be placed on suited background images. We opted for backgrounds from environments that are coherent with the categories of choice. As our case study considers mostly office objects, we tried using either (i) the backgrounds provided with the Washington RGB-D dataset [10] (the ones representing offices/desks), or (ii) a set of backgrounds acquired directly in the robot's operation setting.

To implement this step we exploited the work of [4], which provides methods (with code) to alleviate the artifacts that appear when an object is pasted onto a different background. To augment the size and variability of the final data collection, we applied also the provided set of transformations (2D rotations, scaling, occlusions, etc.) with the addition of illumination changes (brightness, contrast and saturation).

Training. In all our experiments we rely on a convolutional neural network pre-trained on the object categorization task of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [2], specifically, the Caffe [8] implementation[4] of ResNet-50 [6].

We adopt two different transfer-learning methods in the two settings considered in this paper. When targeting object categorization on a benchmark dataset, we fine-tune the network on the generated synthetic training sets. This leads to adapt the image representation to the considered task, but also to the synthetic domain. Since we do not apply sophisticated computer graphics, in this setting we evaluate the trade-off between the benefit of gathering semantically rich images at no cost, and the domain shift possibly introduced in the network by the lack of realism. Differently, in the robotic application we opted to fix the image representation to the one learned on ImageNet and use the synthetic image sets to train linear SVMs on top. We call this on-the-fly learning, because fine-tuning the network takes several minutes/hours, while the training time of the linear SVMs is of the order of seconds and can be interactive.

When fine-tuning, we relied on standard Caffe protocols and just ensured that the learning rate policy was leading to convergence. We used a validation set to stop the training when we observed no accuracy gain and in any case no later than 30 epochs. For each experiment we performed three fine-tuning trials, averaging the results and observing around 1% of performance oscillation across trial. The code for training SVMs employs the *liblinear* [5] package. In this case, cross-validation for the regularization parameter was performed once on an example task.

4 Results

In this section we report on the experimental evaluation that we performed to asses the feasibility of the proposed pipeline for data generation. In all the experiments we use synthetic images generated with our pipeline for training and

[4] `https://github.com/KaimingHe/deep-residual-networks`

test on real images acquired by recording from the camera of humanoid robots. In section 4.1, we present a quantitative evaluation of the performance benefit of each step of the pipeline. To this end, we fine-tune ResNet-50 from ImageNet to address an object categorization task on the iCubWorld dataset [15] which, being recorded from the camera of a robot (iCub[5]) while this is observing hand-held objects, provides a faithful benchmark for real robotic operation settings. In section 4.2, we show that the proposed data generation can also be performed on a robot on-the-fly, to quickly train linear SVMs on specific object categories asked by a user. To this end, we report qualitative results of the pipeline deployment within an interactive object learning application running on the R1 robot[6].

4.1 Benchmark on iCubWorld

In this section we assess if the difference between the synthetic and real domain is such to prevent the suggested approach to be effective for training deep networks. We first report the performance achieved by progressively introducing more processing steps in the proposed image generation pipeline. Then, we show how it is possible to achieve good performance on iCubWorld by injecting a limited amount of real images in a purely synthetic training set.

Real Vs. Synthetic Datasets

Test on iCubWorld. We consider the "iCubWorld Transformations" dataset [15] as our test set (iCWT in the following). This dataset represents 20 object categories of daily use (10 objects per category), each recorded in five image sequences while undergoing isolated viewpoint transformations (e.g., SCALE, 2/3D ROT, BKG, etc.). Each sequence is acquired in two sessions with little setting variations and comprises around 150 frames. We refer the reader to [15] for details. In all experiments, we target a 20-class categorization task and consider, as test set, 5 object instances per category (out of the 10 available) in the BKG sequence. In this sequences the objects are moved by the operator around the robot, keeping their face fixed thus making only the background change. We randomly sampled 50 frames from each sequence, hence our test set is composed of 5K images.

Synthetic Training Sets. To address this task, we downloaded from the Web 80 images for each category in iCWT (see section 3). We randomly selected 60 images per category for training, for a total of 1200 images. The remaining 20 images per category have been used as validation set. After passing the images through foreground segmentation, we remained with around 1150 training images (it is possible that the network used [7] is not able to detect the object, returning an empty mask that is discarded). We tried using two different image sets for the following background replacement step. The first one is from

[5] `http://www.icub.org/`
[6] `https://www.youtube.com/watch?v=TBphNGW6m4o`

the background images publicly available in Washington RGB-D dataset [10]. Specifically, we selected around 350 background images in tabletop-like settings. The second one is a set of the same size, but recorded in the acquisition setting of the iCWT dataset. As explained in section 3, since we relied on the data augmentation procedure from [4], we optionally applied scale, in-plane rotation and light augmentation while replacing the background. In this data augmentation step, for each source image we generated 4 synthetic images, producing a total of around 4600 training images.

Table 1: Classification accuracy achieved by performing diverse processing steps on the downloaded images.

Training Set	**Number of Images**	**Accuracy [%]**
Chance Level	-	5
Web	1200	22
White	1150	18
RGBD	1150	32
iCWT	1150	35
RGBD$_+$	**4600**	**42**

Reducing The Domain Shift - Part I

Ablation Experiment. We evaluated the performance achieved after applying each step of the data generation pipeline and report results after fine-tuning the network on each of the following "intermediate" datasets:

- Images downloaded from the Web (Web).
- Web images segmented with background replacement. We tried a white background (White) or the background from either Washington RGB-D or the iCWT settings (RGBD or iCWT).
- Web images segmented with background replacement and data augmentation. In this case we opted for using the background from Washington RGB-D dataset, since the goal of this work is to build training sets fully automatically and without the need for the user to acquire any data in the operation setting (RGBD$_+$).

We report results in Table 1. We see that just by downloading 60 images per category from the Web we are able to achieve 22% accuracy (chance level is 5%). As expected, the white background replacement is detrimental for performance, while replacing backgrounds which are similar to the one of the test set, does improve results. In this case, the *exact* same background of the test set (iCWT) provides higher results (35%) than one which is similar (RGBD, 32%). However, it is interesting to observe that the performance difference is small, motivating

our choice. A relatively high accuracy (42%) is achieved by applying the data augmentation from [4] to the RGBD training set. However, we note that we are still far from achieving perfect performance. This result establishes a baseline achieved with simple image processing steps. On the one hand, this proves the effectiveness and the potential of the approach. On the other hand, it shows that a better covering of the domain shift is critical to improve performance.

Table 2: Classification accuracy achieved by performing diverse processing steps on the downloaded images (like in Table 1) and by adding real images from iCWT.

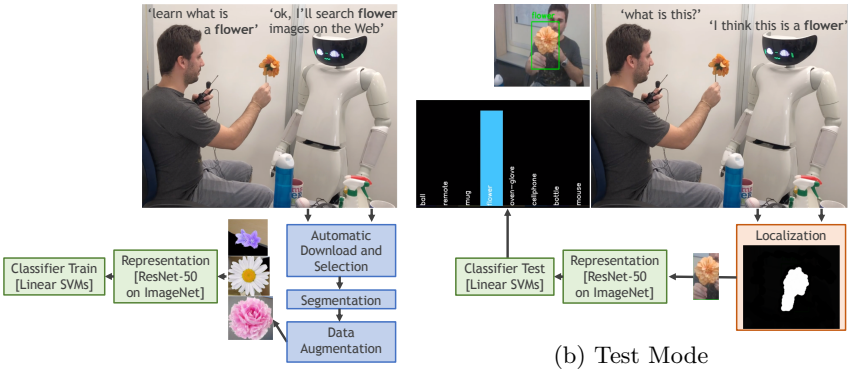
Training Set	**iCWT Objects**	**Number of Images**	**Accuracy [%]**
Web	-	1200	22
Web	1	1200 + 120 (10%) real	46
$RGBD_+$	-	4600	42
$RGBD_+$	1	4600 + 600 (13%) real	**65**
RGBD	1	1150 + 600 (52%) real	63
iCWT	1	1150 + 600 (52%) real	62
Web	1	1200 + 600 (50%) real	62
White	1	1150 + 600 (52%) real	62
$RGBD_+$	5	4600 + 600 (13%) real	73
$RGBD_+$	5	4600 + 5K (109%) real	85
-	5	5K (100%) real	87

Reducing the Domain Shift - Part II

Injection of Real Images. Given the above results, two options can be considered in order to increase performance: (i) improving the realism of the synthetic images and/or (ii) considering the injection of a small set of real images. We opted to evaluate this second possibility, following the suggestion of the authors [4].

In this experiment, we evaluate the performance achieved by adding, to the synthetic training sets considered in the previous section, images of objects from iCWT. For the addition, we sample objects from the remaining 5 instances per category (excluding the test set). Results are reported in Table 2 for two sets of experiments.

We started considering the addition of a single object example per category. We hence sampled from iCWT a few images for each of 20 objects not in the test set (from BKG sequences) and added them to the Web and $RGBD_+$ training sets. We kept the real to synthetic ratio around 10% and used $6\times20=120$ real images for the Web and $30\times20=600$ real images for the $RGBD_+$. In rows 1-4 of Table 2, we observe around 23% performance increase in both cases, achieving 65% with the $RGBD_+$ training set.



(a) Training Mode (b) Test Mode

Fig. 2: Block diagram of the application running on the robot: (a) Training Mode, images are downloaded from the web and used to train the SVMs on the new category; (b) Test Mode, the robot localizes the closest object in the scene and classifies it.

To further investigate to which extent this result depends on the real or synthetic data, in rows 5-8 of Table 2 we increased the real to synthetic ratio up to around 50%, by considering the same 600 real images but adding them to the not augmented synthetic training sets. We observed that performance was almost the same, independently on the quality of synthetic images. It is interesting to note that the addition of as few as 30 frames of a single example instance per category provides a performance gain increase of 40% (from 22 to 62%) when using just Web images. A similar gain is achieved by combining data augmentation (from 22 to 42%) and injection of even less real images (from 42 to 65%).

We finally added all available 5 object examples per category in iCWT to the synthetic training sets (rows 9-11 in Table 2). In this case, with 6 real images per object (for a total of 6×5×20=600 images) we achieve 73% accuracy. Furthermore, adding to the training set as much real images as the synthetic ones (50 frames per object for a total of 5K images) leads to 85% accuracy. This error rate is probably dominated by the information available in the real object examples. This is confirmed by the 87% accuracy achieved by training only on the real images.

4.2 On-the-fly Learning of Object Categories

In this section we briefly describe the deployment of the data generation pipeline to an application running on the R1 robot platform.

The current object recognition system on R1 is the same as the one on the iCub robot and is based on a deep neural network for feature extraction (ResNet-50 trained on ImageNet classification task) and shallow classifiers that

are trained on-the-fly to learn the objects shown by a user [14]. Learning on images acquired during the robot's operation allows for flexibility. However, while this works well for object identification (the robot can observe objects from varied viewpoints) it is time consuming for object categorization [15].

Integrating this data generation pipeline offered an improvement in this direction. To teach a category (Fig. 2(a)), the user tells the label to the robot; the system produces a synthetic training set, that is used to train on-the-fly a classifier. It takes no more than one or two minutes to download and process around 100 images on a standard laptop and internet connection. Images are then encoded into ResNet-50's representation and used to train an SVM (linear Kernel). The feature extraction and classifier training are fast and part of the usual learning pipeline employed on the robot. As showed in the benchmark in Sec. 4.1, the synthetic dataset can also be integrated with real example images acquired by the robot autonomously (see, e.g., our previous work[7]). After training, the robot recognizes the category (Fig. 2(b)). A simple depth segmentation [16] localizes a region of interest as the closest object in the scene, which is then classified.

A video showing qualitatively the performance of the running system is available here[8]. The pipeline deployed on the robot applies data augmentation over background images from Washington RGB-D (the video shows also the kind of noise affecting the content or the foreground masks of the generated images).

The code of the application can run on a normal laptop and can be made publicly available upon request at the same GitHub repository of the original application[9].

While we do not have yet a quantitative benchmark for this data generation pipeline within the on-the-fly training strategy adopted on the robot, we plan to perform such evaluation. Specifically, it would be interesting to compare with prior work [15], where it was shown that, for object categorization in absence of enough object examples, classifier training on top of ImageNet features was more effective than fine-tuning.

5 Conclusions and Future Work

We have studied an automatic pipeline to create task-specific training sets for object categorization. We built the pipeline by downloading "image class seeds" from the Web and composing publicly available code blocks to apply standard image processing, i.e., figure-ground segmentation and blending onto contextual images. This approach is useful in those situation in which example objects are difficult to obtain, as in the case of a robotic system.

Our results showed that simple image processing and data augmentation remarkably improve the performance of the object recognition system (20%) and demonstrated that an additional performance gain (40%) can be obtained

[7] `https://youtu.be/HdmDYIL48H4`
[8] `https://youtu.be/eIb9GjIOYXo`
[9] `https://github.com/robotology/onthefly-recognition`

by integrating the synthetic dataset with a small set of real images of a similar object, taken from the robot. This is interesting, because such a set could also be used to disambiguate the web research, by providing a visual example together with the category label.

The approach presented in this paper can potentially be extended to other tasks in robotics, in which fast adaptation is hampered by the cost of acquiring training samples. For example, in future work we plan to address object detection tasks, by combining our recent work [12]. In this perspective, this line of research could be key to develop vision systems trainable on-the-fly on novel categories.

Acknowledgements

B. C. acknowledges the financial support of the Project ERC RoboExNovo.

References

1. D. S. Cheng, F. Setti, N. Zeni, R. Ferrario, and M. Cristani. Semantically-driven automatic creation of training sets for object recognition. *Computer Vision and Image Understanding*, 131:56–71, 2015.
2. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc CVPR*, 2009.
3. S. K. Divvala, A. Farhadi, and C. Guestrin. Learning everything about anything: Webly-supervised visual concept learning. In *Proc CVPR*, 2014.
4. D. Dwibedi, I. Misra, and M. Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proc ICCV*, 2017.
5. R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.
6. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc CVPR*, 2016.
7. S. D. Jain, B. Xiong, and K. Grauman. Pixel objectness. *arXiv*, 1701.05349, 2017.
8. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proc. ACM*, pages 675–678. ACM, 2014.
9. A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc NIPS*, 2012.
10. K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Proc ICRA*, 2011.
11. Y. LeCun, Bernhard E. B., J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *Proc NIPS*, 1990.
12. E. Maiettini, G. Pasquale, L. Rosasco, and L. Natale. Speeding-up object detection training for robotics with falkon. In *Proc IROS*, 2018.
13. N. Massouh, F. Babiloni, T. Tommasi, J. Young, N. Hawes, and B. Caputo. Learning deep visual object models from noisy web data: How to make it work. In *Proc IROS*, 2017.

14. G. Pasquale, C. Ciliberto, F. Odone, L. Rosasco, and L. Natale. Teaching icub to recognize objects using deep convolutional neural networks. In *Proc ICML Workshop on Machine Learning for Interactive Systems*, volume 43, 2015.
15. G. Pasquale, C. Ciliberto, F. Odone, L. Rosasco, and L. Natale. Are we done with object recognition? The iCub robot perspective. *Robotics and Autonomous Systems*, 112:260 – 281, 2019.
16. G. Pasquale, T. Mar, C. Ciliberto, L. Rosasco, and L. Natale. Enabling depth-driven visual attention on the icub humanoid robot: Instructions for use and new perspectives. *Frontiers in Robotics and AI*, 3:35, 2016.
17. M. Samadi, T. Kollar, and M. Veloso. Using the web to interactively learn to find objects. In *Proc AAAI*, 2012.
18. M. Waibel, M. Beetz, R. D'Andrea, R. Janssen, M. Tenorth, J. Civera, J. Elfring, D. Gálvez-López, K. Häussermann, J. M. M. Montiel, A. Perzylo, B. Schießle, O. Zweigle, and R. van de Molengraft. RoboEarth - A World Wide Web for Robots. *Robotics & Automation Magazine*, 18(2):69–82, 2011.
19. 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 *Proc ECAI*, 2016.