Learning Accurate Objectness Instance Segmentation from Photorealistic Rendering for Robotic Manipulation

Siyi Li[1], Jiaji Zhou[2], Zhenzhong Jia[2],
Dit-Yan Yeung[1], and Matthew T. Mason[2]

[1] The Hong Kong University of Science and Technology
{sliay,dyyeung}@cse.ust.hk
[2] Carnegie Mellon University
{jiajiz,zhenzjia,matt.mason}@cs.cmu.edu

Abstract. Recent progress in computer vision has been driven by high-capacity deep convolutional neural network (CNN) models trained on generic large datasets. However, creating large datasets with dense pixel-level labels is extremely costly. In this paper, we focus on the problem of instance segmentation for robotic manipulation using rich image and depth features. To avoid intensive human labeling, we develop an automated rendering pipeline for rapidly generating labeled datasets. Given 3D object models as input, the rendering pipeline produces photorealistic images with pixel-accurate semantic label maps and depth maps. The synthetic dataset is then used to train an RGB-D segmentation model by extending the Mask R-CNN framework for depth input fusion. Our results open up new possibilities for advancing robotic perception using cheap and large-scale synthetic data.

Keywords: instance segmentation, robotic manipulation, synthetic data

1 Introduction

Over the past two decades, robotic solutions have been increasingly deployed in various domains, including households, industrial manufacturing, warehouse automation, and so on. However, it is still challenging for robots to perform complex tasks autonomously in unstructured and unknown environments. Grasping and manipulating [1] individual items in cluttered and constrained space is such a problem remaining largely unsolved. Consider the process of picking up all the objects from a cluttered bin, the objects could be arranged with partial or even full occlusion. Heavy stacking of several homogeneous objects also poses non-trivial challenges for the robot to determine the correct number of those instances. These issues must be addressed from the perception side.

Remarkable progress has been made by utilizing deep learning to perception problems in recent years. Those promising results are typically obtained by training Convolutional Neural Networks (CNNs) using an extensive amount of

labeled data. Though there exist unsupervised methods such as convolutional autoencoders to reduce the required labeled training samples, the performance of these approaches is less satisfying. Therefore a key question is how to create such large datasets with minimal human labor.

Recently an increasingly popular approach is to avoid manual annotations by using synthetic data generated by simulators. While labeled data are difficult to obtain in the real world, they are very easy to generate in the simulator. It has been shown that CNNs trained in synthetic data exhibit reasonable generalization performance on several perception tasks such as object detection and pose estimation [2, 3]. This opens up new promises for transferring deep perception models trained in simulators to real-world robotic scenarios.

In this paper we study perception for robotic manipulation in cluttered environments. The perception capability to segment each object instance is crucial for successful manipulation. We focus on the instance segmentation problem, which requires the detection of all objects in an image while also precisely segmenting each instance. We develop an automated rendering pipeline to generate labeled datasets for training CNNs. Our pipeline takes 3D CAD models as input and produces photorealistic labeled images with enough variety. We train an RGB-D objectness instance segmentation system by extending Mask R-CNN [4] to leverage depth sensor information. Through extensive experiments, we show that models trained on photorealistic synthetic data allow for accurate object instance segmentation in real-world manipulation scenarios.

2 Related Work

Object detection and segmentation has been dramatically improved in recent years. For object detection, one successful line of work considers the task as region proposal followed by classification and scoring. The so-called Region-based CNN (R-CNN) framework [5, 6] has evolved much both in terms of accuracy and efficiency. The success of semantic segmentation dates back to the advent of Fully Convolutional Networks (FCNs) [7]. The recent Mask R-CNN [4] extends R-CNN framework by adding a small FCN branch for predicting segmentation masks and is currently the state of the art for object instance segmentation. To utilize deep learning in robotic vision, most existing work attempts to make use of the real-world data in an efficient manner. These methods either set up a self-supervised training pipeline in real world [8] or apply transfer learning by using a small amount of real-world samples [9].

Synthetic data has been extensively used in computer vision. Modern open-world computer games have been used to create perception benchmark [10]. 3D object models are also increasingly used for training neural networks for object detection [11], human pose estimation [12] and viewpoint estimation [13].

Synthetic data has also been applied to robotic vision. Sadeghi *et al.* [14] learn a policy for real-world quadrotor obstacle avoidance by using a simulator with varied 3D scenes and textures. [15] demonstrates that the behavior of a high-level control policy trained in a physics simulator can be transferred to

real-world quadrotor for autonomous target following. In [2], the authors propose a self-supervised learning system for object detection, where the initial model is pre-trained using synthetic images and fine-tuned over real-world images. [3] explores the domain randomization technique in pose estimation by training only in simulated images with non-realistic random textures.

3 Synthetic Data Generation

Figure 1 shows the pipeline for synthetic data generation. Our rendering system is based on Blender [16]. Our 3D CAD model database is a subset of YCB object dataset [17] containing 20 objects. For each scene, we first choose a subset of objects from the model database with random initial poses above the bin. Then the Bullet physics [18] engine simulates the objects falling into the bin by gravity before settling down into a stable equilibrium state. The final poses will then be rendered with the specific setting of lighting condition and camera view to generate the final RGB output.

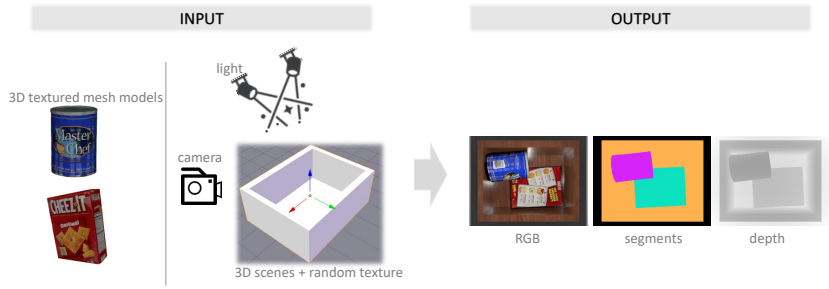


Fig. 1: Our pipeline for generating synthetic data.

Fig. 2: Sample RGB images from our generated dataset.

Object poses: We maintain a set of dropping positions relative to the bin to make sure that object will mostly drop into the bin. The orientations are sampled from $\{(0,0,0),(\pi/2,0,0),(0,\pi/2,0),(0,0,\pi/2)\}$ to cover all possible views.

Lighting and camera: We use point lighting source, which is varied with respect to location and intensity to create shadows and over exposures. The projective camera has a resolution of 640×480, focal length of 35mm. The camera position is sampled from a pyramid on top of the bin center.

Domain randomization: We also domain randomize the background bin. In particular, the random textures are chosen from random RGB values or sampled from online material texture resources. Fig. 2 shows some samples.

4 Learning Objectness Instance Segmentation

Our instance segmentation system is based on Mask R-CNN [4]. We develop several techniques to leverage Mask R-CNN with depth input for instance segmentation in the manipulation scenario.

Objectness training: Instead of performing multi-class instance segmentation, here we only care about objects versus non-objects. Therefore we utilize *objectness* Mask R-CNN to distinguish objects from non-objects (background). Intuitively, learning the notion of objectness instance makes it easier for the model to generalize to novel objects unseen in the training set.

Model architecture: Our model uses the ResNet-50 [19] with Feature Pyramid Network (FPN) [20] as the convolutional backbone due to its accuracy and efficiency. The original Mask R-CNN only takes an RGB image as input and produces a set of detection bounding boxes and mask maps. To make full use of RGB-D cameras, we extend the Mask R-CNN framework to leverage depth information. Figure 3 shows the proposed multi-modal network architecture, where both the RGB input and the depth input (replicated across channels to match the RGB) are fed into the shared ResNet-50 tower. Then the feature maps are concatenated across channels, followed by an additional convolutional layer to merge the features. The rest of the network remains the same. One advantage of this design is that both the RGB input and depth input can utilize available ResNet-50 backbone pre-trained in ImageNet [21] dataset. The whole network is then fine-tuned for the segmentation task.

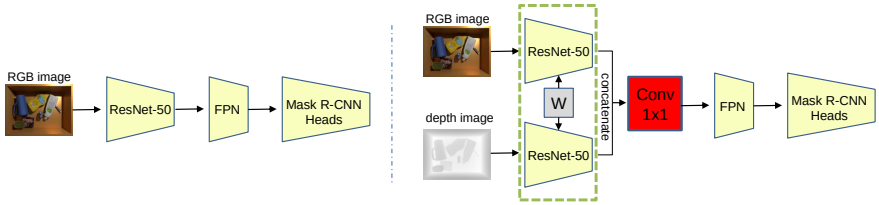


Fig. 3: An illustration of the original Mask R-CNN architecture (left) and the proposed RGB-D Mask R-CNN architecture (right).

5 Multi-view Real-World Data Generation

We also develop an automated approach to efficiently generate real-world labeled data by leveraging robots with wrist mounted cameras. We program the robot to take color and depth images from multiple views by generating camera poses in a pyramid on top of the table center with a gazing constraint. For each scene, the robot takes more than fifty views of images, five of which are manually labeled. We then use a correspondence algorithm described below to label each point with consistent instance identity (ID) from the separate three views in the fused point cloud. Finally, we perform projection of the labeled point cloud onto each unlabeled view and generate segmentation masks for each of them. Figure 4 shows the pipeline of the fusing algorithm.

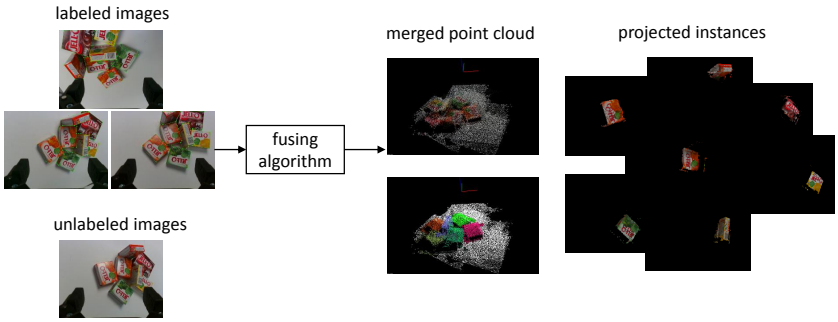


Fig. 4: Multi-view label fusing pipeline.

Multi-view instance matching: For two different labeled views, the two point clouds corresponding to the same instance usually do not share the same label ID. We assume that for the labeled views, the corresponding point clouds of the same instance viewed from different camera poses still share a certain amount of overlapping area. This can be then used to decide instance correspondence and merging, as described as follows.

1. We construct an occupancy grid out of the existing merged point cloud (initialized from the first point cloud): each grid g_{ijk} stores the points that belong to the grid and the label of the grid l_{ijk} is determined as the majority label among all points in the grid.
2. To merge a new labeled view, for each point p in the point cloud PC with label l that is mapped to a grid g_{ijk} with label l_{ijk}, we then add the overlapping count among label ID l in the new view with l_{ijk} in the existing aggregate point cloud. These counts between labels of the new view and labels of the existing views form a correlation matrix.
3. We use best weighted bipartite graph matching treating the correlation matrix as a weight matrix and merge the points and labels according to the matching result.

6 Experiments

In this section, we present a series of experiments to evaluate performance of the proposed objectness instance segmentation system. We first conduct extensive ablation studies on synthetic data to validate both the data generation design choices and the capability of the Mask R-CNN model. Then we evaluate the full system on the real-world dataset generated by the multi-view instance matching algorithm.

6.1 Implementation Details

We train the model using images of size 640×480 without resizing. We train on 2 GTX 1080Ti GPUs for 100k iterations, with a learning rate of 0.025 which is reduced by 10 times at the 40k iteration. We use Synchronized SGD with a weight

decay of 0.0009 and a momentum of 0.9. Other hyper-parameters are set according to the original paper [4]. To leverage features trained from a larger image domain, the convolutional backbone ResNet-50 is pre-trained on ImageNet [21] for 1000-way object classification. The whole model is then fine-tuned on our generated synthetic dataset to learn general objectness instance segmentation.

6.2 Evaluation Metrics

We use the commonly average precision (AP) metric. Given an image, the model outputs a number of mask predictions sorted by confidence values. A mask prediction is considered as positive if its IoU (intersection-over-union) with the ground truth is above a certain threshold. By choosing different number of predictions, it gives a plot of the precision against the recall values. AP can be viewed as finding the area under the precision-recall graph. We report the COCO-style metrics [22] including AP_{50} (AP over IoU 0.5), AP_{75} (AP over IoU 0.75) and AP (average AP over IoUs ranging from 0.5 to 0.95 with a step of 0.05).

6.3 Data Generation Design Choices

With the ability of rapidly generating a large amount of labeled data, one may ask questions of "how much data is needed?" and "which type of data is most valuable?". To that end, we compare the following six different strategies for the data generation process:

a) **single-object**, where only one object instance is rendered for each image
b) **multi-object**, where multiple object instances are rendered for each image
c) **multi-object with varying texture**, where multiple object instances are rendered with varying background textures for each image
d) **multi-object with varying lighting**, where multiple object instances are rendered with varying lighting conditions for each image
e) **multi-object with varying camera views**, where multiple object instances are rendered with varying camera perspective views for each image
f) **multi-object with varying all**, where multiple object instances are rendered with varying all the above axes for each image.

For each of the strategy, we render a training set of 500 images and train a model with RGB input only. For strategy a), we render 2500 images to roughly match the number of instance annotations. Those models are then evaluated on a common test set with 500 images generated with all variations.

Table 1 left compares the performance of different models trained with different training sets on the common test set. We first observe that the most crucial aspect of the data generation process is to have multiple objects in the scene. If the model only sees single-object scenes in the training process, it can hardly generalize to heavily occluded and cluttered multi-object cases. Furthermore, we see that domain randomization on the proposed three axes all consistently improves the generalization performance of the trained model and hence all of them are necessary to generate a dataset with enough variety.

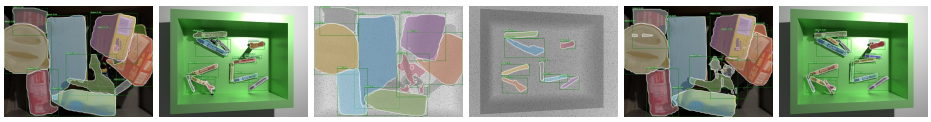
training data	a	b	c	d	e	f
AP	0.89	21.26	34.80	33.03	36.57	**46.78**
AP_{50}	0.99	36.97	57.98	51.47	58.02	**70.66**
AP_{75}	0.99	21.62	36.10	35.96	39.25	**50.49**

input	RGB	depth	fusion
AP	74.13	69.63	**81.35**
AP_{50}	89.49	87.54	**92.59**
AP_{75}	80.12	74.95	**85.38**

Table 1: Left: instance segmentation mask AP results on the common test set of models trained using different data generation strategies. Right: instance segmentation mask AP on the synthetic test set of models with different inputs. Units are percentage numbers.

6.4 Evaluation on Synthetic Data

To evaluate the capability of the objectness Mask R-CNN model on synthetic images, we formally create a larger training set consisting of 8000 images with varying background texture, lighting and camera views. A test set consisting of 2000 images is rendered separately. We compare Mask R-CNN trained on RGB only, Mask R-CNN trained on depth only and Mask R-CNN trained on both inputs. Table 1 right shows the results. We find that when only using single input, Mask R-CNN performs significantly better when trained on RGB data, indicating that in most cases the color textures provide more discriminative information than the geometric structure. Moreover, RGB and depth fusion with joint training achieves the best performance. Depth signal proves to be very useful for extreme cases such as very dark lighting condition or heavily occluded small objects. For example, we can observe from Figure 5 that the fusion method usually constructs better-quality masks and reliably detects small object parts even when they are heavily occluded. Also the high AP values in Table 1 indicate that the proposed segmentation system can almost perfectly fit the synthetic data.



RGB only depth only fusion

Fig. 5: Qualitative results of models with different inputs on two synthetic scenes.

6.5 Evaluation on Real-World Data

To construct the real-world dataset, we collect both the RGB and depth images under 10 different manipulation scenes using the wrist mounted camera. The ground-truth masks are then generated semi-automatically by the multi-view instance matching algorithm in Section 5. The final dataset contains about 600 images. Figure 6 shows the object configurations of the different manipulation scenes.

For training, we render a synthetic dataset consisting of 10000 images with random object configurations from the 20 objects of the 3D database. Note that the training object set is only a subset of the real-world objects. For example,

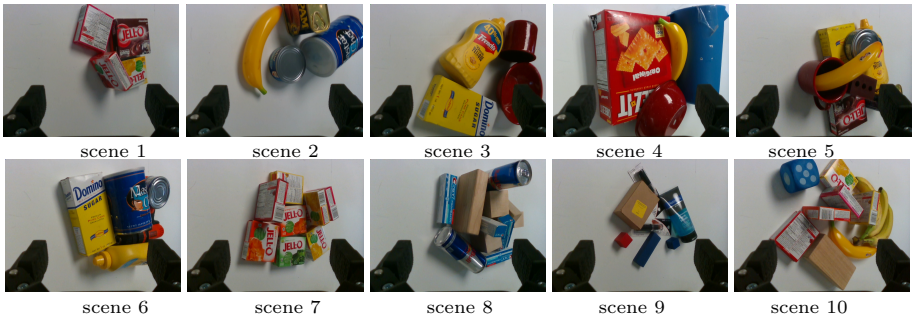


Fig. 6: Object configurations of the multi-view real-world data.

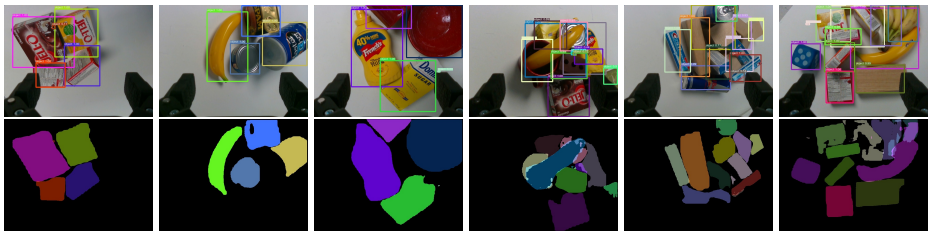


Fig. 7: Qualitative results of RGB-only model on real-world scenes.

scenarios 8-10 in Figure 6 all contain novel objects not in the training object database, including Red Bull bottles, Colgate toothpaste boxes, real bananas and so on. This allows us to further evaluate the generalization ability of the proposed segmentation system.

To overcome the data bias between synthetic data and real-world data, we take the following data augmentation procedures. For the RGB data, we propose to paste real-world background images to the background part of the original synthetic image. This adapts the model to be resistant to real-world distractors such as the gripper in the image corner. For the depth data, we add random noise to the perfect synthetic depth image since real-world depth sensor observations are usually noisy. Besides the aforementioned data augmentation, we do not use any labeled real-world data in the training phase.

scene	1	2	3	4	5	6	7	8	9	10	all
AP	**66.46**	**53.37**	**78.40**	51.50	**36.30**	**56.87**	**30.48**	37.40	**42.26**	**42.31**	**39.11**
	65.73	49.98	77.56	**52.20**	27.62	49.06	24.50	**37.54**	31.19	38.20	34.81
AP_{50}	84.84	**84.17**	99.46	74.63	**65.43**	**79.30**	**56.70**	**63.95**	**67.13**	**69.49**	**65.72**
	88.82	77.42	**100**	**86.15**	53.50	74.09	46.83	61.31	53.45	58.06	59.12
AP_{75}	76.24	**50.22**	**83.54**	**51.84**	**34.18**	61.99	**27.02**	41.95	**43.24**	**42.86**	**39.43**
	76.35	42.42	77.39	46.15	25.54	**62.03**	24.51	**45.16**	25.57	42.17	34.72

Table 2: Instance segmentation mask AP results of RGB-only (upper row) and fusion (lower row) models on real-world data. Units are percentage numbers.

To compare the generalization ability of models with different inputs, we evaluate both the RGB-only model and the fusion model on the real-world dataset. Table 2 shows the quantitative results of the two models on the presented 10 manipulation scenes. We can observe that the overall AP scores of the two models both decrease compared to the performance on the synthetic dataset. This is reasonable since the models are directly transferred without any adaptation. Nevertheless, both models can still achieve a AP score over 30, which can be considered to be pretty usable in most manipulation scenarios. From the per-scene AP results, we also find that both models achieve high accuracy for known object cases (scene 1-6), indicating that the proposed photorealistic rendering pipeline minimizes the gap between simulator and reality. Scene 5 and 7 are two challenging highly cluttered cases where different objects are heavily stacked together. Thus our models achieve significantly lower accuracy compared to the simple cases. Note that scene 7 is harder since two adjacent homogeneous objects may be falsely detected as one, making it difficult to determine the correct number of instances. As for the novel objects cases (scene 8-10), our model exhibits very good generalization ability. Most of the instances can be correctly detected and segmented even with occlusion. Figure 7 further shows the representative qualitative results.

For the real-world evaluation, we observe that the RGB-only model performs better than the fusion model on most cases. This is mainly due to the noisiness of the depth information from the depth sensor. The real-world depth images usually contain holes due to some reflective part of the objects. Although we try to add simple random noise in the data augmentation process, the gap between different depth signals still exists. Trying to simulate the real-world depth noise in the rendering process is a possible extension to alleviate this problem.

7 Conclusion and Future Work

In this paper, we have explored the potential of utilizing synthetic data for real-world robotic perception scenarios. We develop an automated rendering pipeline to generate photorealistic images with pixel-level semantic annotations given the scanned 3D CAD models. Large datasets can be easily collected without any human intervention. We propose an objectness instance segmentation system by extending the Mask R-CNN framework for multi-modal input fusion. Extensive experiments show that the proposed system exhibits good generalization performance from synthetic data to real-world scenarios. For future work, we plan to build a larger 3D object database and further improve the capability of the proposed system.

Acknowledgments

This research has been supported by General Research Fund 16207316 from the Research Grants Council of Hong Kong.

References

1. Zhou, J., Paolini, R., Johnson, A.M., Bagnell, J.A., Mason, M.T.: A probabilistic planning framework for planar grasping under uncertainty. IEEE Robotics and Automation Letters **2**(4) (2017) 2111–2118
2. Mitash, C., Bekris, K.E., Boularias, A.: A self-supervised learning system for object detection using physics simulation and multi-view pose estimation. In: IROS. (2017)
3. Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world. In: IROS. (2017)
4. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV. (2017)
5. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014)
6. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NIPS. (2015)
7. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. (2015)
8. Zeng, A., Yu, K.T., Song, S., Suo, D., Walker Jr, E., Rodriguez, A., Xiao, J.: Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In: ICRA. (2017)
9. Schwarz, M., Milan, A., Periyasamy, A.S., Behnke, S.: Rgb-d object detection and semantic segmentation for autonomous manipulation in clutter. The International Journal of Robotics Research (2016)
10. Richter, S.R., Hayder, Z., Koltun, V.: Playing for benchmarks. In: ICCV. (2017)
11. Pepik, B., Stark, M., Gehler, P., Schiele, B.: Multi-view and 3d deformable part models. IEEE transactions on pattern analysis and machine intelligence **37**(11) (2015) 2232–2245
12. Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M.J., Laptev, I., Schmid, C.: Learning from synthetic humans. In: CVPR. (2017)
13. Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In: ICCV. (2015)
14. Sadeghi, F., Levine, S.: $(CAD)^2$RL: Real single-image flight without a single real image. In: RSS. (2017)
15. Li, S., Liu, T., Zhang, C., Yeung, D.Y., Shen, S.: Learning unmanned aerial vehicle control for autonomous target following. In: IJCAI. (2018)
16. Blender. `http://www.blender.org/`
17. Calli, B., Singh, A., Walsman, A., Srinivasa, S., Abbeel, P., Dollar, A.M.: The ycb object and model set: Towards common benchmarks for manipulation research. In: ICAR. (2015)
18. Bullet. `http://bulletphysics.org/`
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)
20. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR. (2017)
21. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3) (2015) 211–252
22. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: ECCV. (2014)