Learning to Segment Affordances

Timo Lüddecke, Florentin Wörgötter
{tluedde, worgott}@gwdg.de
University of Goettingen, Faculty of Physics, Computational Neuroscience Group

Abstract

The goal of this work is to densely predict a comparatively large set of affordances given only single RGB images. We approach this task by using a convolutional neural network based on the well-known ResNet architecture, which we blend with refinement modules recently proposed in the semantic segmentation literature. A novel cost function, capable of handling incomplete data, is introduced, which is necessary because we make use of segmentations of objects and their parts to generate affordance maps. We demonstrate both, quantitatively and qualitatively, that learning a dense predictor of affordances from an object part dataset is indeed possible and show that our model outperforms several baselines.

1. Introduction

Most computer vision tasks, such as object recognition or optical flow, aim at an accurate description of what is there. In this domain, we witnessed remarkable progress in recent years, driven be the success of deep learning techniques. But is this always the right objective? The well-known psychologist J.J. Gibson postulates that the primary purpose of vision in humans and other animals is to serve biological needs [8]. Instead of obtaining a complete and accurate picture of a situation, we are rather concerned about possible interactions with the environment, e.g. "Where can I drink from?" or "Where can I leave this dish?". Gibson designates such relations between human (and animal in general) and their surrounding with the term *affordances*, as they indicate what the environment affords.

Considering scenes in terms of possible actions or affordances instead of object labels is compelling not only for humans but also for machines. In real life scenarios, for instance in robotics, where action is always demanded, the space of affordances is a more natural than that of object labels. There is no guarantee that you can sit on a chair no matter how good you are at detecting them, e.g. it might be occupied or put on the table. A street is not walk-able because we call it a street but because it is a planar surface orthogonal to gravity. Thus learning a proper model for sitable areas, even if it is nothing but a cardboard box, turns out to be a more effective strategy to master reality.

The latter represents the key idea of this paper: We hypothesize that all relevant cues required to predict affordances can be estimated from the image using local structure, global context or experience. Evidence for walkability could be provided locally by a homogeneous texture and in the context of cars parking on it or trees flanking this surface. Even invisible, yet crucial cues, like weight or rigidity, can be reliably guessed from images if they are seen often enough. In order to investigate this hypothesis, in this work, we design a method (see Figure 1) that extracts affordance maps from object-part segmentations and trains a convolutional neural network to densely predict these affordances from a single RGB image.

Focusing on affordances is also interesting from a theoretical perspective: Both, object labels and affordance labels, impose different divisions of the abstract space of all possible image segments. However, not all divisions are equally good (i.e. semantically similar segments should be grouped together). The split provided by affordances might particularly encourage generalization as affordances are often associated with particular geometric or contextual features such that corresponding classes become more meaningful.

Contributions To the best of our knowledge, this is the first work to employ object part segmentations specifically to learn affordances. This enables us to teach a set of 15 well-defined affordances to our model, which is considerably larger than that in prior works. Since our training data is fragmentary, we introduce a novel loss function compensating for the incomplete data by regarding the coverage of valid pixels. The learned network carries out inference at almost 10 frames per second, enabling robotic applications.

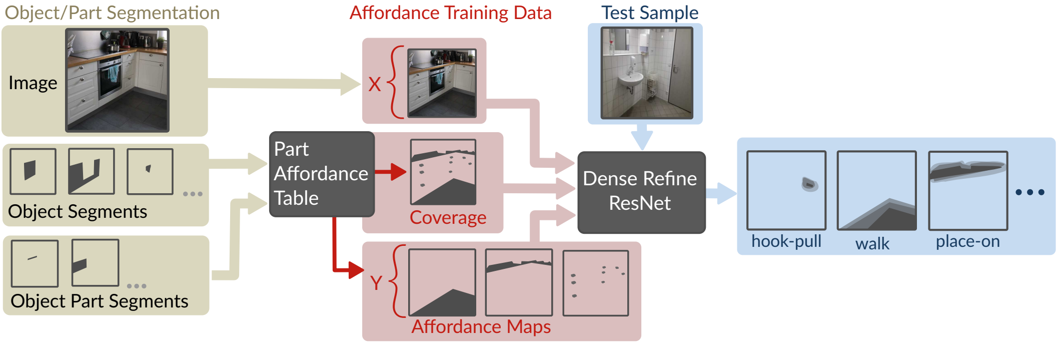

Figure 1. Our approach: We train a neural network to predict a set of affordance maps (Y) from a single RGB image (X) using a new loss function that allows for incomplete data by incorporating a coverage map.

2. Related Work

Dense Labeling Since we label each pixel of the image our approach falls in the domain of dense prediction. It is similar to semantic segmentation, where the goal of pixel-wise predictions is shared but the predicted classes are commonly exclusive. In this area, like many other within computer vision, neural networks have become the dominant tool. A common strategy is to employ a network that is pre-trained for whole-image labeling, remove the classification layers at the top, add further layers which are capable of up-sampling to output an image and retrain the altered network on a densely labeled dataset. Long, Shelhamer, and Darrell [14], use the VGG16 [21] network as a basis to densely predict class labels by averaging up-sampled "skip" branches. In the DeepLab model, Chen et al. [3] additionally apply a conditional random field to achieve a better alignment between the model's predictions and the image's edges. Eigen and Fergus [5] do not only predict object class labels but also depth and normals using an architecture that alternates between convolutions and up-sampling incorporating skip connections at multiple scales. The dilated convolutions by Yu and Koltun [26] avoid reduction of spatial accuracy during pooling while still banking on the same pre-trained classification network. Badrinarayanan, Kendall, and Cipolla [1] address the same problem by storing the pooling indices and passing them to the up-sampling layers.

In object segmentation, which aims at generating masked object proposals, Pinheiro et al. [16] introduced refinement modules, which successively merge high- and low-level information and combined them with ResNet architecture [11].

Affordances Viewing the scene from a functional perspective, akin to the notion of affordances, has a long tradition in computer vision, with early, rule-based approaches, dating back more than 30 years [23]. More recent methods globally calculate affordances for entire objects [22] [28]. Ye et al. [25] detect bounding boxes of affordances using a two-stage approach consisting of region proposal and CNN-feature-based affordance recognition. Often affordances are linked with corresponding poses. Gupta et al. [10] first estimate the scene geometry and then infer from it a set of four affordances while Grabner, Gall, and Van Gool [9] only considers sit-able locations in images. Similarly, Fouhey et al. [7] use video to predict affordance maps. Yao, Ma, and Fei-Fei [24] differentiate modes of interaction between humans and objects by examining the pose depicted in images. The method of Myers et al. [15] densely labels 7 affordances in images of tools using RGB-D data.

A similar concept to affordance segmentation are "action maps", with the difference that actions can be very specific, such as "using a laptop" while affordances not necessarily involve concrete objects. Savva et al. [19] analyse RGB-D video recordings and track people to generate 7 "action maps" while Rhinehart and Kitani [17] recently made use of egocentric video in order to learn maps for 6 actions.

Arguably the most related work to our approach has been suggested by Roy and Todorovic [18]. They share the goal of predicting affordances per-pixel from a single RGB image. Intermediate representations for depth, normal and semantic segmentations are obtained and used to derive five types of affordances, which are differently defined than ours. A dataset used for training and evaluation involving RGB, intermediate maps and affordances is created based on NYUv2 dataset [20] using a semi-automatic procedure with manual correction.

In contrast to the presented methods, our approach predicts a larger number of affordances using a novel training method that explicitly makes use of part information. We focus on directly predicting affordances without incorporation any hint of scene geometry. A comparison of closely related approaches is provided in Table 1.

Approach	#	input	output
Grabner, Gall, and Van Gool [9]	1	RGB-D	per voxel
Gupta et al. [10]	4	RGB	per-pixel
Savva et al. [19]	7	Video	per voxel
Rhinehart and Kitani [17]	6	Video	per grid-cell
Roy and Todorovic [18]	5	RGB	per pixel
our approach	15	RGB	per pixel

Table 1. Comparison of related algorithms with # denoting the number of used affordances.

object	obstruct	pinch-pull	break	sit	grasp	illumination	support	place-on	...
*/knob	1	1	0.5	0	1	0	0	0	...
*/top	0	0	0	0.5	0	0	0	1	...
pot	1	0	0.5	0	1	0	0	0	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

Table 2. Excerpt from the transfer table.

3. From Object Part Labels to Affordances

In this section we describe how to leverage annotations on the object and object-part level to generate a large set of pixel-wise maps for our 15 types of affordances shown in Table 3. To obtain the latter set we follow three guiding principles:
(a) Affordances should refer to interactions that can be *valuable* (in any context) for robots or humans. (b) Action names are often highly imprecise in the sense that continuous movements differ vastly within the same action. For example, consider "put into", which could imply a large set of trajectories. We approach this problem by selecting and defining our affordances to imply only a *specific trajectory*, where this is possible. E.g. we opt for "pinch-pull" instead of "open" as the latter is unspecific and might imply different trajectories. (c) The same action can invoke affordances of different object-parts. E.g. a house is enter-able, a door (which is a part of the house) is open-able and the door's handle (as a part of the door) is pull-able. Along this hierarchy, we opt for the most *specific* level, which is pull-able in this case.

Object Parts Affordance segmentation is a challenging problem, since we only want to label the fraction of a surface that actually offers an affordance. Since affordances refer to surfaces and media, they do not necessarily correspond one-to-one to objects. In fact, only a small fraction of an object might provide an affordance. For instance, consider a chair: Only the seating surface affords to sit, while arm- and backrest enable support. A standing lamp is a sizable object but only its bulb actually provides light. Although we say "open a cabinet", this refers to pulling on a small handle on its door. A key observation of this work is that affordances tend to pertain to single parts of objects rather than objects as a whole. This motivates us to put an emphasis on object parts within the context of this work. While common segmentation datasets [12], [20], [6] [13] only provide segmentations for whole objects, the recently introduced ADE20K dataset [27] actually resolves objects into their parts. Thus, we decide to use this particular dataset in our work.

Transfer Table The mapping from object labels to affordance maps is conducted using a look-up table. Although it is manually defined, we try to keep the amount of required labor minimal by automatizing the process as much as possible. As exemplified in Table 2, in the transfer table, either object names like "cabinet", paths to object-parts like "cabinet/drawer/handle" or pure parts like "*/drawer" are associated with different 15-dimensional affordance vectors, with each dimension corresponding to one affordance. For each object or part, multiple affordances can be present simultaneously. To transform a concrete object or part to an affordance the object-part path is searched from specific to general. This means in the example above, we first assess if "cabinet/drawer" is specified in the table and only if it is not found "*/drawer" is visited. We consider the 500 most frequent objects and parts from the dataset and only provide affordance vectors if the affordances can be reliably attributed to the respective object or part. This applies in particular to very large objects like "cabinet" that offer multiple affordances but none of them being valid for the whole object. As a consequence, for some parts of the image, no affordances are provided. During training these cases are carefully handled in a way that is described in section 4.1, exploiting the fact that we know where data is missing. By allowing for this type of incompleteness we are able achieve a high precision, i.e. if annotations are provided then they are correct, at the cost of a reduced recall, i.e. not all correct annotations are included. Simply put, we prevent our network from being confronted with too much wrong data.

Data Augmentation The scene quality within the dataset varies a lot. Some scenes are captured and annotated in high resolution and resolve objects into many fine-grained parts while others have significantly fewer pixels, are blurry and describe objects as a whole. Also, the number of training samples generated from ADE20K is comparatively low given that we employ a data-driven approach. Consequently, we augment the dataset by sampling multiple cropped image patches from an original image and slightly vary color and contrast of these patches. The number of crops is dependent on the original image quality: We sam-

Affordance	**Description**
obstruct	vertical surface that prevents locomotion. *e.g. wall*
break	detachable objects that can easily be damaged or destroyed *e.g. vase*
sit	surface a human can sit on while having the feet on the ground *e.g. seat cushion*
grasp	detachable objects that can be encompassed with one hand or only few fingers and be moved with one arm.*e.g. vase)*
pinch-pull	surfaces that can be pulled through a pinch movement (all directions). *e.g. knob*
hook-pull	surfaces that can be pulled by hooking up fingers (all directions). *e.g. handle*
tip-push	surfaces that trigger some action when being pushed. *e.g. button-panel*
warmth	surfaces that emit warmth. *e.g. fireplace*
illumination	surfaces that emit visible light.*e.g. bulb*
observe	surfaces that present information or art, i.e. that can be read or watched. *e.g. display*
support	stable surfaces that provide support for standing (for the agent) except ground. *e.g. wall*
place-on	raised surfaces where objects can be placed on (this excludes the ground). *e.g. tabletop*
dry	surfaces capable of soaking water. *e.g. towel*
roll	surfaces that can be used with wheels. *e.g. road*
walk	surfaces a human can walk on. *e.g. grass*

Table 3. Description of the set of affordances being addressed in this work.

ple more crops if the image is large and contains many objects.

4. CNN-based model

The scene affordance task at hand demands contextual integration while making crisp predictions. The former means that the receptive field of each pixel should be as large as the image itself, as even remote pixels can be crucial for the local affordance. This can be exemplified by the decision whether a surface is walk-able or suitable to place things (place-on-able): Locally, both the ground and a table surface, appear as flat, uniform areas but by taking the context into consideration this affordance ambiguity can be resolved. Walk-able surfaces tend to be flanked by cars and trees while we rather expect chairs to accompany place-on-able surfaces.

As a second requirement, details should not get lost during the forward pass of the network such that relevant boundaries in the image are preserved. This is particular important because some of our affordances commonly refer to tiny and thin structures, e.g. the hook-pull-able handles. If the network's predictions are too coarse, they can not be properly identified.

In the work of Pinheiro et al. [16], ResNet50 and refinement modules were successfully combined to generate object proposals. This is not unexpected since ResNet, pre-trained with ImageNet weights [4], is one of the best-performing methods in image classification while refinement modules offer an elegant way to merge local with scene-level information. Since model requirements are akin between object segmentation and affordance segmentation, we adopt [16] as a core architecture and modify it for our purposes as described below.

Refinement Module The refinement modules which we employ as illustrated in Figure 2 are inspired from [16] but simplified, omitting a convolutional layer. They integrate abstract information from deep layers with less deep layers, which tend to encompass spatially accurate information. Both input layers must deliver maps of the same image size. First they are stacked on top of each other (concatenated along depth), then convolved to obtain $15k$ feature maps, with k being a hyper-parameter of our model that will be assessed in Section 5. Since our training dataset is comparatively small, we only train the refinement modules, while preserving the weights learned from ImageNet in the original ResNet.

4.1. Cost function

While formulating a cost function we encounter the challenge of dealing with the incomplete data which is obtained through converting object parts to affordance maps. In this case, incomplete means, that some regions of the target prediction are invalid. Here, in contrast to defined regions, we can not tell whether an affordance is present or not, because the corresponding object or part is not found in the transfer table.

However, since we assembled the affordance maps, we know the location of the invalid regions. The idea is to expose this information to the cost function. Another important trait of affordance segmentation is that affordances are not exclusive, hence cost functions common in semantic segmentations cannot be employed here as they assume for each pixel a probability distribution over a set of object classes. Contrarily, we imply a binary (present vs not present) probability distribution for each pixel and each affordance. Regarding both aspects, we propose a novel cost function we call masked binary cross entropy. Subsequently, we will formally derive it.

We denote the ground truth matrix of an image for affordance $a \in \mathcal{A}$ and pixel $i \in \mathcal{I}$ with $\mathbf{Y}_{ai}$ and the associated model prediction with $\hat{\mathbf{Y}}_{ai}$. Furthermore, the binary cross entropy H is defined by: $H(p,q) = -p\log(q) - (1-p)\log(1-q)$. This is integrated into a scalar loss or cost, which describes the average binary entropy over all affor-

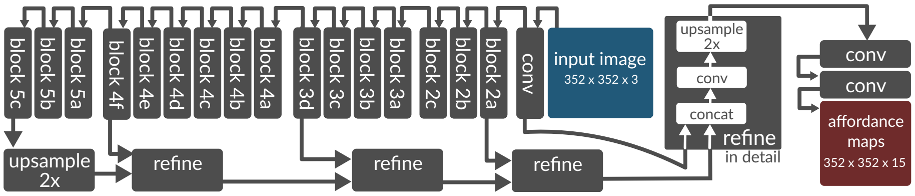

Figure 2. Architecture of our proposed Dense-Refine-ResNet. A ResNet50, which is illustrated block-wise is extended with Refinement modules. All refinement modules are structured like the one shown in detail.

dances and the image.

$$\mathcal{L}(\mathbf{Y}, \hat{\mathbf{Y}}) = (|\mathcal{A}||\mathcal{I}|)^{-1} \sum_{a\in\mathcal{A}} \sum_{i\in\mathcal{I}} H(\mathbf{Y}_{ai}, \hat{\mathbf{Y}}_{ai})$$

While the binary cross entropy loss is compatible with non-exclusive classes, it does not account for incomplete data yet. To do so, we suggest a simple but effective step: Mask the cross entropy matrix before averaging yielding the following loss:

$$\mathcal{L}^{\mathrm{m}}(\mathbf{Y}, \hat{\mathbf{Y}}) = (|\mathcal{A}| \sum_{i\in\mathcal{I}} \mathbf{M}_i)^{-1} \sum_{a\in\mathcal{A}} \sum_{i\in\mathcal{I}} \mathbf{M}_i H(\mathbf{Y}_{ai}, \hat{\mathbf{Y}}_{ai})$$

with $\mathbf{M}_i \in \{0, 1\}$ indicating if pixel i is valid, i.e. if a corresponding entry is found in the transfer table.

5. Experiments

To quantitatively assess the models, we use intersection over average (denoted by *IoU*) and accuracies (*Acc*) as measures, which are common in semantic segmentation. For both we can compute scores per affordance class and average them (*mean*) or ignore class boundaries and apply the measures pixel-wise (*pixel*). Before presenting results, we report the experimental set-up in detail.

Evaluation Datasets Since we are not directly training on the ADE20K dataset but transform the object and object part labels to affordances and the test set is not publicly available we divide the datasets slightly differently: Our training samples are generated (as described in Section 3) from a 90% portion of the ADE20K *training* set with the remaining 10% being used to derive our validation set. The evaluation is conducted on two different sets, both being initially generated from the ADE20K *validation* set: A dataset of 432 scenes, which we call LG432, is used to assess how well the models generalize on the distribution they have been trained on. Due to the cost of annotation, during training, affordance maps derived from object parts (as described above) serve as a proxy for the true distribution of the affordance maps. However, eventually we are interested in how well the true distribution is learned. This quality is estimated using the HQ50 dataset, which comprises 50 high quality scenes (high resolution, many objects being present) annotated by an expert according to the definitions listed in Table 3. Hence, it represents the true distribution.

	LG432			HQ50
	mean	pixel	pixel	mean
	IoU	IoU	Acc	IoU
k=7	**32.3%**	**63.0%**	**75.6%**	28.9%
k=5	31.1%	62.2%	74.8%	28.5%
k=3	32.2%	63.0%	75.3%	**30.0%**
k=1	29.3%	60.2%	71.3%	28.5%
224px / k=5	30.4%	59.7%	71.8%	27.9%
vgg_upconv	28.4%	58.1%	70.3%	26.1%
vgg_refine	22.7%	49.1%	59.5%	24.9%

Table 4. Comparison of different hyperparameters and other architectures. Image size is 352px if not indicate otherwise.

test dataset	LG432	HQ50
break	36.2%	42.2%
dry	25.3%	10.9%
grasp	21.8%	25.0%
hook-pull	00.0%	16.0%
illumination	39.1%	50.2%
obstruct	73.9%	59.3%
pinch-pull	00.0%	00.3%
place-on	14.6%	08.8%

	LG432	HQ50
read/watch	29.4%	16.7%
roll	70.5%	71.7%
sit	06.6%	07.7%
support	69.8%	66.4%
tip-push	00.9%	03.6%
walk	70.5%	71.5%
warmth	23.6%	00.0%
mean	**32.2%**	**30.0%**

Table 5. Individual intersection over union scores for the affordances in both test datasets. Note that pinch_pull and hook_pull are not covered by LG432 and hence set to zero.

Output Binarization The metrics described above require the model predictions to be binary. Since our network naturally predicts real numbers expressing their certainty for the presence of an affordance, the original output is thresholded. The thresholds are determined on a 20% subset of the LG432 test set which is subsequently spared for the actual evaluation.

Implementation Details Training is carried out on a Geforce Titan X and took several hours for each model. Early stopping with patience of 2 is used. All models are trained using RMSprop with a learning rate of 0.001 using keras with Theano backend [2].

5.1. Model Comparison

Quantitative scores of our models are reported in Tables 4 and 5. Mean IoU mostly ranges around 30 % while the pixel-wise measures yield larger numbers. This is due to our networks performing best for classes which cover large fractions of the image (e.g. walk or obstruct) and therefore having a stronger impact than rare classes (e.g. tip-push) on the pixel-wise measures. Unbalanced class frequency also partially explains deviations between LG432 and HQ50: If a class occurs less often, its IoU and accuracies variance is increased.

Determining the best configuration of our model, we observe that larger values of k tend to cause better scores on LG432, although there is a small drop for $k = 5$. However, when evaluating against the true (human defined) distribution represented by HQ50, performance peaks at $k = 3$. For an explanation, note that a larger value of k leads to a bigger number of parameters. Hence, it is possible that the complex models ($k > 3$) overfit to the artifacts introduced by the object-part to affordance conversion, while a lower complexity of $k = 3$ enables better generalization. Consequently, $k = 3$ turns out to be the favorable model, which comes with the additional advantage of being quite fast compared to the other, more complex models, with an average processing time for an image of 107 ms. Large input image sizes are beneficial for performance, which can be seen for $k = 5$. While this is the only pair involving different image sizes reported in the table, we actually conducted more experiments, all of which support this assumption. This is not surprising, though, as larger images capture more detail, which can be used to assess the presence of affordances, in particular small structures like handles or knobs.

We also compare to VGG-driven [21] baseline models. These involve a simple encoder-decoder architecture, which uses VGG16 activations after the last convolutional layer and alternatives between convolution and up-sampling until the original image size is retained (*vgg_upconv*). The *vgg_refine* model is similar to our ResNet-based network but uses VGG16 as a basis and different values for k. *vgg_dilated* refers to a model akin to [26]. Note, in all cases our novel cost function is used. All of these reference models are outperformed by our network in all configurations. Compared to semantic segmentation an IoU of 32.2 % is rather low, e.g. Long, Shelhamer, and Darrell [14] achieve a mean IoU of around 62 % on the Pascal VOC dataset [6]. However, our task is more difficult since affordances are not exclusive, i.e. certainty on the presence of an affordance does not necessarily diminish the probability of other affordances. Additionally, semantic segmentation models profit from large, manually corrected datasets, which are not available to us.

5.2. Affordance-wise Evaluation

We assess the model's performance for each individual affordance class and report scores in Table 5. Hook_pull and pinch_pull do not occur in the LG432 test dataset and therefore yield a score of zero. The IoU scores confirm good performance in frequent categories, like walk or obstruct, being present in almost every image. For rare affordances, performance is weaker, though. But note that the test sets are comparatively small, which (a) complicates the derivation of proper thresholds of rare affordances and (b) makes it very difficult to properly asses such rare affordances, as tiny variations can have a large impact on the reported performances. Remarkably, in some categories, e.g. illumination, break or grasp, the performance on HQ50 is higher than on LG432, which provide evidence for the initial hypothesis of affordances being highly suitable for generalization.

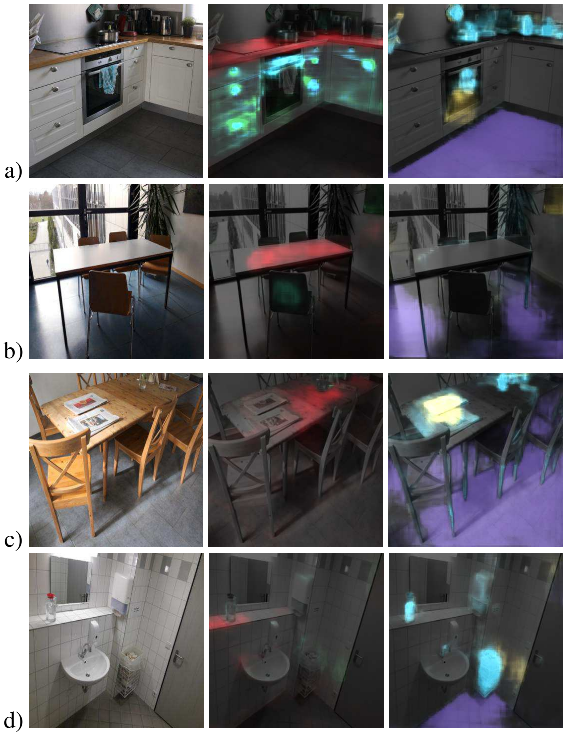

Figure 3. Various scenes with corresponding predictions from our network. Left: Original images. Middle: Place-on (red), pinch-pull(blue), hook-pull(green). Right: walk (purple), grasp (blue) and read/watch (yellow).

5.3. Qualitative Samples

The kitchen depicted in a) encompasses many knobs and appliances, which without exception are correctly labeled. Also, the towel in front of the oven is considered to be grabable. Along the border of the cabinet door we can even identify hints of hook-pull-ability. In image b), the front-lighting impedes scene understanding. Nonetheless, almost the entire table surface is identified as place-on-able while the most of the ground is properly labeled walk-able. The newspaper on the living room table in c) are considered to be both, grasp-able and observable, which is true. In the artificially illuminated scene in image d) the shelf is only partially recognized as place-on-able while the non-detachable waste bin is considered grasp-able. However, the glass bottle is rightly identified as grasp-able, although it is transparent. After all, the qualitative evaluation indicates that our method works well in many cases, but some situations causing problems, possibly due the training data not being generic enough.

6. Conclusion

We introduce a novel approach to affordance segmentation building on the insight that affordances often pertain to object parts. It is shown that the model is able to pixel-wise label a fairly large set of affordances - sometimes even in daring situations. Particularly, we present a simple, yet effective way how to harness incomplete segmentations to build a model that carries out complete predictions. A fast inference speed of only 107ms makes it suitable for robotic applications. We are confident that better performance could be obtained by more and higher-quality part segmentations datasets, for example by a dataset that decomposes a larger set of objects into fine-grained parts.

Prospectively, predictions from our network could serve as a prior for a curious robot. By interacting with the environment the model could continuously improve, being free from the dependency on manually collected datasets one day.

Acknowledgements The research leading to these results has received funding from the European Community's H2020 Programme (ICT-26-2016) under grant agreement no. 731761, IMAGINE.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Scene Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).

[2] James Bergstra et al. "Theano: A CPU and GPU math compiler in Python". In: *Proc. 9th Python in Science Conf.* 2010, pp. 1–7.

[3] Liang-Chieh Chen et al. "Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs". In: *ICLR*. 2015.

[4] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2009*. IEEE, 2009, pp. 248–255.

[5] David Eigen and Rob Fergus. "Predicting Depth, Surface Normals and Semantic Labels With a Common Multi-Scale Convolutional Architecture". In: *The IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015.

[6] Mark Everingham et al. "The pascal visual object classes challenge: A retrospective". In: *International Journal of Computer Vision* 111.1 (2015), pp. 98–136.

[7] David F Fouhey et al. "People watching: Human actions as a cue for single view geometry". In: *International journal of computer vision* 110.3 (2014), pp. 259–274.

[8] James J. Gibson. *The Ecological Approach to Visual Perception*. Houghton Mifflin, 1979.

[9] Helmut Grabner, Juergen Gall, and Luc Van Gool. "What makes a chair a chair?" In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011, pp. 1529–1536. (Visited on 10/16/2014).

[10] Abhinav Gupta et al. "From 3D Scene Geometry to Human Workspace". In: *Computer Vision and Pattern Recognition(CVPR)*. 2011.

[11] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[12] Tsung-Yi Lin et al. "Microsoft COCO: Common Objects in Context". English. In: *European Conference on Computer Vision (ECCV)*. Ed. by David Fleet et al. Vol. 8693. Springer International Publishing, 2014, pp. 740–755. ISBN: 978-3-319-10601-4.

[13] Ce Liu, Jenny Yuen, and Antonio Torralba. "Sift flow: Dense correspondence across scenes and its applications". In: *IEEE transactions on pattern analysis and machine intelligence* 33.5 (2011), pp. 978–994.

[14] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

[15] Austin Myers et al. "Affordance Detection of Tool Parts from Geometric Features". In: *ICRA*. 2015.

[16] Pedro O Pinheiro et al. "Learning to Refine Object Segments". In: *ECCV*. 2016.

[17] Nicholas Rhinehart and Kris M Kitani. "Learning action maps of large environments via first-person vision". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 580–588.

[18] Anirban Roy and Sinisa Todorovic. “A Multi-scale CNN for Affordance Segmentation in RGB Images”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 186–201.

[19] Manolis Savva et al. “SceneGrok: Inferring action maps in 3D environments”. In: *ACM transactions on graphics (TOG)* 33.6 (2014), p. 212.

[20] Nathan Silberman et al. “Indoor segmentation and support inference from RGBD images”. In: *European Conference on Computer Vision*. Springer. 2012, pp. 746–760.

[21] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *arXiv preprint arXiv:1409.1556* abs/1409.1556 (2014).

[22] Michael Stark et al. “Functional object class detection based on learned affordance cues”. In: *International conference on computer vision systems*. Springer. 2008, pp. 435–444.

[23] Patrick H. Winston et al. “Learning physical descriptions from functional definitions, examples, and precedents”. In: *Proeccedings of AAAI*. 1983.

[24] Bangpeng Yao, Jiayuan Ma, and Li Fei-Fei. “Discovering object functionality”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 2512–2519.

[25] Chengxi Ye et al. “What Can I Do Around Here? Deep Functional Scene Understanding for Cognitive Robots”. In: *To appear at ICRA* abs/1602.00032 (2017).

[26] Fisher Yu and Vladlen Koltun. “Multi-Scale Context Aggregation by Dilated Convolutions”. In: *ICLR*. 2016.

[27] Bolei Zhou et al. “Semantic understanding of scenes through the ade20k dataset”. In: *arXiv preprint arXiv:1608.05442* (2016).

[28] Yuke Zhu, Alireza Fathi, and Li Fei-Fei. “Reasoning about Object Affordances in a Knowledge Base Representation”. In: *Computer Vision - ECCV 2014*. Springer, 2014, pp. 408–424. (Visited on 10/16/2014).