Detecting hands with a convolutional neural network trained using synthetic data

Barbara Aljaž, Luka Čehovin Zajc

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko
E-pošta: ba4918@student.uni-lj.si, luka.cehovin@fri.uni-lj.si

Abstract

Deep learning, more precisely convolutional neural networks, are known for the large amount of data they require for training. One of the new approaches of providing a sufficient amount of training samples is their artificial generation using computer graphics. In this paper we present a use-case of training a convolutional neural network detector using generated images of hands. We have evaluated the detector on a dataset of real images for a touch-less interface human-computer interaction scenario and compared it with a detector trained on real images. Results of the experiment are promising and present many opportunities for further development of such training technique.

1 Introduction

Deep learning, more specifically, the use of multi-layer convolutional neural networks (CNN) has made a significant impact in the computer vision field in the last decade, performance on tasks like detection, recognition, classification, and tracking has been greatly improved. This was primarily made possible by new methodological discoveries, but also by the rapid increase in computing power and storage capacities, as well as availability of data.

But despite its abundance, collecting suitable data for various domains is challenging and expensive. Training and testing images or videos have to be collected, filtered, and annotated. Collected images may also include bias and may not represent the appearance variations uniformly. Since deep learning methods are notably data-hungry with training sets well above 1000 samples, a need for easier and more flexible ways of data collection has emerged. Beyond data augmentation that increases dataset size using simple image processing operations (affine transformations, blur, noise), artificially generated samples can be used as well in some cases. Modern computer graphics approaches can produce realistic images of objects. Moreover, the properties of an object and its surroundings can be controlled during data generation process, reducing bias and giving full control over annotation process. More importantly, the process is automatic, meaning that we can generate large quantity of training samples and quickly adapt to new learning scenarios.

In this paper we investigate the role of synthetically generated data in training a CNN-based detector for human hands. This kind of detector can be used as a component in various human-computer (HCI) and human-robot (HRI) interaction scenarios. Hands are also very suitable test domain for such approach because of high DoF and reasonably low variability between different hands. We have developed a system that generates a large amount of synthetic training images based on a parameterized 3D model of a hand and combine them with an arbitrary background, as shown in Figure 2. These samples were then used to train a CNN detector that we have evaluated on a dataset of real images of hand poses.

2 Related work

Hand localization and gesture recognition has a long history of research. Traditionally, color cues were used [1] since such approach is computationally effective, but the performance of such method suffers in case of color similarity or difficult lighting conditions. With the advent of low-cost depth cameras (e.g. Kinect), hand detection has become easier in constrained usage scenarios and some gestures could be detected with simple image processing operations [2]. More recently, machine learning has also been used, e.g. AdaBoost [3], Random forest [4], and even more recently CNNs [5, 6]. All these approaches relied on datasets of images of real hands which are challenging to build in an unbiased manner and difficult to extend to different training scenarios.

On the other side, synthetic imagery has been successfully utilized before in various, predominantly HCI, scenarios. In one of the initial attempts generated depth images were used to train a random-forest-based body-part detector [7], and more recently to determine hand posture parameters using CNNs [8]. These methods achieve good performance also due to lower amount of appearance variation of depth images in comparison to color images.

Computer-generated color images have been utilized in deep learning to do 3D face reconstruction [9] and to detect various objects in an autonomous driving scenario [10]. Recently several attempts have used generated hand images for hand pose recognition. In [11] authors used synthetic images to classify pre-localized hand images into several gestures classes. In [12], a full hand

pose is recovered from a pre-localized hand. In [13], the approach is extended into temporal domain. In contrast, our presented work is focused on solving hand localization using synthetic images which could be seen as complementary step to posture recognition.

3 The hand detector

Our process of creating a real-time hand detector consists of generation of artificial training samples, described in Section 3.1, training a suitable CNN network, described in Section 3.2.

3.1 Dataset generation

For synthetic data generation we have used an open-source 3D model [1], seen in Figure 1. The model consists of 2590 faces, 59 bones, and has skin-like texture enhanced using sub-surface scattering technique. We have used a Python script within the Blender 3D modelling application [2] to generate images with random skeleton configurations, hand global rotation and color hue as well as position and intensity of environmental lights.

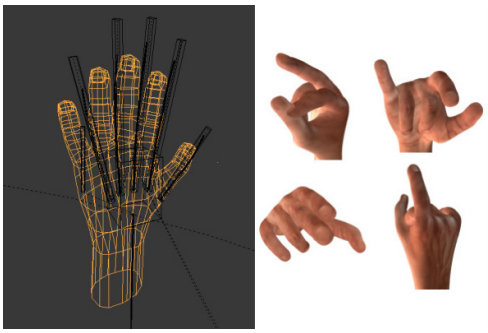

Figure 1: Wire-frame representation of a hand model (left) and some generated poses rendered on transparent background (right).

For the current experiments we have generated 100 different hand poses with 20 instances of each, some examples are shown in Figure 1. All images were rendered with transparent background. In the next step we have combined hand images with random real images that were verified that they did not contain visible hands. Background images were randomly cropped and the overlaying hand images were augmented with combination of flipping, rotating, blurring as well as shifting hue and saturation components. Some examples of generated learning images are presented in Figure 2.

3.2 Detector

For the detector have used a modified YOLO network architecture [14] that is known for its real-time performance. The architecture is different from commonly used R-CNN because it uses a single network for extracting potential foreground candidates and their categorization. The network is consequently trained on an entire image

[1] free3d.com/3d-model/rigged-hands-28855.html
[2] www.blender.org

Figure 2: Examples of artificially generated images used in learning process with bounding boxes that mark groundtruth.

and can therefore get more information about the context around the object. The network architecture is inspired by the GoogLeNet model [15], it has 24 convolutional layers followed by two fully connected layers. The learning process and inference consist of dividing an image into a grid where every cell can predict a certain number of bounding boxes, their scores and class probabilities. The loss function is composed of 5 parts - error of predicted bounding boxes' coordinates, error of their dimensions, error of object presence, error of object absence and classification error. This approach makes inference very fast, but it is also prone to having lower detection accuracy and recall because of a limited number of potential bounding boxes.

We have started from the original YOLO architecture used for general object detection. We have lowered the number of filters in first fully connected layer from 4096 to 1024 since our problem is easier and we can sacrifice some complexity in order to achieve higher inference speed, the final architecture is presented in Figure 3. We have also added batch normalization layers used during training. We have trained the network from scratch and, based on our preliminary experiments, used Adam optimizer with more than two hundred times smaller learning rate in comparison to the original network, to avoid divergence during training. The training was performed using batch size of 5 and trained for a maximum of 100 epochs.

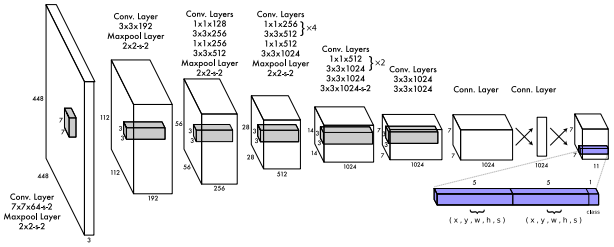

Figure 3: Network architecture of our detector.

Since detector can provide multiple detections for a single object, we perform a non-maxima suppression step

to reduce the number of overlapping proposals. We have clustered all detection with mutual overlap of 0.1 together and retained only the one with the highest detection score.

4 Experimental evaluation

In our evaluation we have focused on using a detector to detect hands in camera-facing HCI scenario. The parameter distribution of hands was therefore adapted for such view-point. We have evaluated our detector on a hand gesture dataset [16, 17], that consists of RGB, depth images, and information from a Leap Motion sensor (in contrast to other work that uses the dataset we have used only RGB images because of the nature of our study). The dataset contains 1400 images of 14 different people showing 10 different gestures with 10 samples per gesture. Some examples of test images can be seen in Figure 4. Since bounding-box annotations were not provided, we annotated images using LabelImg [3] utility and cropped them from the left and right side in order to obtain a square image suitable for the YOLO network (which expects images of 448×448 pixels). In the evaluation we have followed a standard PASCAL evaluation procedure for object detection [18]. We present results for the evaluation in form of precision-recall curves and best F-scores in Figure 5.

Figure 4: Examples from the testing dataset.

We begun our evaluation with a model trained exclusively on synthetic images, denoted as *only_syn*. This model was trained on 5000 synthetic samples. This model achieved F-score of 0.758 when evaluated on the entire testing dataset. The performance of the model did not increased if it was trained on more images, which could signify that the variability of synthetic training data was not sufficient. Indeed, our current data generation approach does not take into account background-foreground contrast which can be observed as a problem in Figure 6. One of the main problems of the model was also poor accuracy of the YOLO architecture, which results in many detection windows with overlap a bit below the 0.5 PASCAL threshold. The shape of the precision-recall plot for the *only_syn* model shows that some wrong detections have very high scores. Upon a closer investigation we have discovered that most of these detections were indeed positioned on a hand, had a good detection score but with a very poor overlap with the groundtruth.

We have also evaluated the model trained on synthetic images on another synthetically generated dataset of 5000 images, we denote this result as *syn_on_syn*. The model achieved better performance with F-score of 0.88. This was expected, but the score is still low enough that

[3] github.com/tzutalin/labelImg

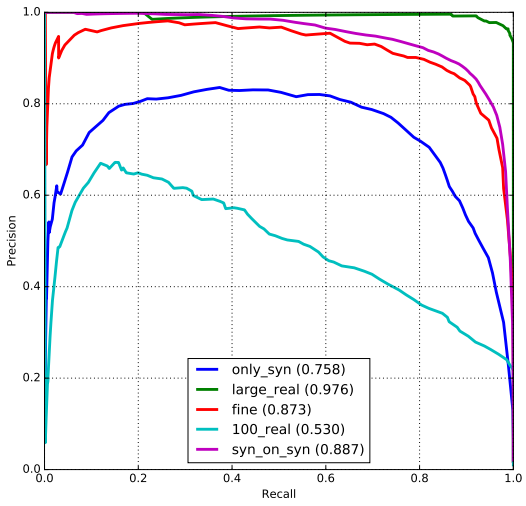

Figure 5: Precision-recall curve. The optimal F-score for each curve is provided in brackets next to the corresponding legend entry.

we cannot talk about over-fitting of the model to the synthetic data and shows potential for mining hard samples for more efficient training.

In order to see the limit of training the same network on realistic data, we have trained the detector from scratch on a random 80% of our realistic dataset (1120 images) and tested on the remaining 20% (280 images). The results are denoted as *large_real*, the best achieved F-score was very high, i.e. 0.97. This was expected since images were chosen randomly and the similarity of 10 instances of the same person showing the same gesture is very high, which means that the model is likely very overfitted. We would also like to emphasize that, in contrast to this detector, that was trained on a selected number of testing gestures, the detector trained on synthetic images is more general and that its performance would probably not degrade even if new gestures would be added to the testing dataset.

The last experiment that we have performed addresses fine-tuning of a model trained with synthetic data with some real samples. Similar approach has been investigated in [11] and was confirmed to improve performance of their model. We have used 100 images (two people showing five gestures) from the realistic dataset to fine-tune the model. The evaluation of the model, denoted as *fine*, was performed on a subset of the testing dataset that excluded both people and all five gestures to avoid overfitting. The best F-score rose to 0.873, a 10% improvement in comparison to detector learned only on synthetic data. A reasonable explanation for this increase could be the role of the context in convolutional neural networks. A strong cue for the presence of a hand could also be a wrist, part of an arm or even face. Since our synthetic images only contain hands the appropriate context is missing and was only introduced during fine-tuning on real

data. We have also trained the model denoted *100_real* only on 100 real images and achieved best F-score of 0.53 on the testing subset. This number is still quite high, which probably means that the testing dataset is not very diverse.

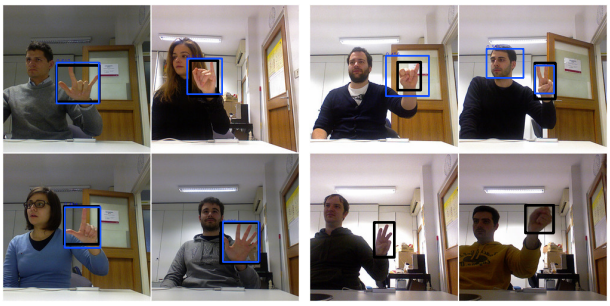

Figure 6: Examples of successful detection (left) and various failure cases (right) that occurred either because of a small overlap with groundtruth, detection of face due to skin color similarity, false negative because of image contrast and false negative due poor lightning conditions and motion blur. Black rectangles represent ground-truth, blue rectangles represent detections.

5 Conclusion

In this paper we have presented our study on suitability of using computer generated synthetic data to train a CNN detector. We have developed a synthetic data generation system for images of human hands, trained a CNN detector and evaluated it on realistic dataset. We have achieved encouraging performance that shows the potential of using synthetic images for deep learning in computer vision.

We have also discovered several potential directions for further work. While the current model of a hand is quite realistic, it does not acknowledge the role of context in CNNs, that could improve detection accuracy. We also intend to extend training dataset with more variability in terms of hand pose and other factors (e.g. hand hue, lights). In terms of accuracy we will evaluate other fast CNN architectures that improve localization. We also plan to evaluate our model on more challenging datasets and compare it with existing approaches. Finally, our system enables effortless extension into recognition of hand postures and well as hand gestures. Such a system could be a building block of many HCI or HRI applications.

References

[1] Xiaojin Zhu, Jie Yang, and Alex Waibel. Segmenting hands of arbitrary color. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 446–453. IEEE, 2000.

[2] Yi Li. Hand gesture recognition using kinect. In *2012 IEEE International Conference on Computer Science and Automation Engineering*, pages 196–199, June 2012.

[3] Mathias Kölsch and Matthew Turk. Robust hand detection. In *FGR*, pages 614–619, 2004.

[4] Xian Zhao, Zhan Song, Jian Guo, Yanguo Zhao, and Feng Zheng. Real-time hand gesture detection and recognition by random forest. In Maotai Zhao and Junpin Sha, editors, *Communications and Information Processing*, pages 747–755. Springer, 2012.

[5] Akshay Rangesh, Eshed Ohn-Bar, and Mohan M Trivedi. Hidden hands: Tracking hands with an occlusion aware tracker. In *CVPRW 2016*, pages 19–26, 2016.

[6] Shiyang Yan, Jeremy S Smith, Yizhang Xia, Wenjin Lu, and Bailing Zhang. Multi-scale convolutional neural networks for hand detection. In *Applied Computational Intelligence and Soft Computing*, volume 2017, 04 2017.

[7] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR 2011*, CVPR '11, pages 1297–1304, Washington, DC, USA, 2011. IEEE Computer Society.

[8] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Training a feedback loop for hand pose estimation. *CoRR*, abs/1609.09698, 2016.

[9] E. Richardson, M. Sela, and R. Kimmel. 3d face reconstruction by learning from synthetic data. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 460–469, Oct 2016.

[10] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV 2016*, pages 102–118. Springer, 2016.

[11] C. J. Tsai, Y. W. Tsai, S. L. Hsu, and Y. C. Wu. Synthetic training of deep cnn for 3d hand gesture identification. In *2017 International Conference on Control, Artificial Intelligence, Robotics Optimization (ICCAIRO)*, pages 165–170, May 2017.

[12] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single RGB images. *CoRR*, abs/1705.01389, 2017.

[13] Franziska Mueller, Dushyant Mehta, Oleksandr Sotnychenko, Srinath Sridhar, Dan Casas, and Christian Theobalt. Real-time hand tracking under occlusion from an egocentric RGB-D sensor. *CoRR*, abs/1704.02201, 2017.

[14] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[16] Giulio Marin, Fabio Dominio, and Pietro Zanuttigh. Hand gesture recognition with leap motion and kinect devices. *2014 IEEE International Conference on Image Processing (ICIP)*, pages 1565–1569, 2014.

[17] Giulio Marin, Fabio Dominio, and Pietro Zanuttigh. Hand gesture recognition with jointly calibrated leap motion and depth sensor. *Multimedia Tools Appl.*, 75(22):14991–15015, November 2016.

[18] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL visual object classes (VOC) challenge. *International journal of computer vision*, 88(2):303–338, 2010.