On Helping Stroke Rehabilitation with Faster R-CNN

Jinwoo Choi[*]
Department of Electrical and Computer Engineering
Virginia Tech
Blacksburg, VA 24061
`jinchoi@vt.edu`

Abstract

In this work, we address the problem of real-time hand detection problem for automatic stroke patient rehabilitation assessment system. We employ the state-of-the-art object detector, Faster R-CNN, to detect hands of various skin-tones and under illumination variations. We got preliminary experimental results of reasonable mean average precision score with real-time test speed in this work. As our future works, we will apply the Faster R-CNN detection to other objects and integrate the detection modules to the stroke patient rehabilitation system.

1 Introduction

In this work, we address the hand detection problem from video for real-time applications. This work is designed with a purpose of application to a bigger project: helping stroke rehabilitation with an autonomous rehabilitation exercise evaluation system. In the system, a stroke survivor performs some pre-defined exercises which consist of grasping some objects, moving them from one place to another, releasing the objects, and manipulating the objects. In order to assess the quality of the exercises without any humans in the loop, the system should detect and track hands, torso, and other objects and it should also recognize which grasp type the survivor uses and assess the quality of the grasp.

Object detection is a fundamental computer vision problem. In order to develop higher level vision algorithms such as grasp detection, action recognition, and event detection which are employed in the stroke rehabilitation system, we need more reliable and accurate detection method. Thus, in this work, we focus on the hand detection problem among all the building blocks of the large system. However, we expect that the hand detection problem addressed in this work can be applied to other detection problems such as torso and other object.

2 Related Works

Li and Kitani proposed a pixel-level hand detection method in [1] in order to apply it to the stroke rehabilitation system. They used random tree regressors with various features (color, texture, histogram of oriented gradients, SIFT, ORB, super-pixel features). In order to account for various skin-tones, background clutter, various poses, and illumination changes, they trained the multiple regressors conditioned on the illumination and they used the mixture of multiple regressors as a final model. Even though they tried to build a model which is robust and accurate, it is still not very robust to the various skin-tones and illumination conditions.

Recently, there have been great advances in the object detection task [2]-[5] and these advances have been driven by convolutional neural networks (CNNs) [6], [7]. In the CNN based detection method, CNN is mainly employed as a classifier. In [2], a region proposal method such as selective search

[*]https://sites.google.com/site/jchoivision/

generates object-like regions. Then CNN classifies each region into categories and this method is called region-based CNN (R-CNN) method. Although R-CNN achieves state-of-the-art detection performance in terms of mean average precision (mAP), it is slow at test-time to be deployed for real-time applications due to many independent forward passes of the CNNs. In [3], they modified the network architecture to share computation of convolutional layers between object proposals and this method is called Fast R-CNN. In [4], which is called Faster R-CNN, the object proposal task is incorporated into the CNNs rather than having an external region proposal module. Fast R-CNN and Faster R-CNN achieve the 25 and 250 times faster test-time speed than R-CNN without loss of detection accuracy respectively. In this work, we employ Faster R-CNN because our goal is apply the hand detection module to the real-time stroke rehabilitation system.

3 Approach

In order to detect hands in each image, we need not only a hand classifier but also a bounding box regressor for hands. In this work, we formulate the hand detection problem as a structured prediction problem. In this formulation we want to predict $[x_{min}, y_{min}, x_{max}, y_{max}]$ which correspond to the top-left and bottom-right coordinates of the hand, for a given image. There are only two classes in this problem definition: hand and background.

Faster R-CNN [4] consists of two networks on top of shared convolutional layers. One network is region proposal network (RPN), another network is Fast R-CNN for object classification. Convolutional layers are shared across the two networks to efficiently train a model and to train the entire model in an end-to-end manner.

In the forward pass, an input image is fed into the shared convolutional layers. The output of the shared convolutional layers are convolutional feature maps. The feature maps are fed into RPN. The proposed regions are fed into Fast R-CNN detector along with convolutional feature maps. Then the Fast R-CNN module classifies each region. Detailed illustration of the Faster R-CNN algorithm is described in [4].

We follow the 4-step training algorithm in [4].

1) **RPN training**: Initialize the RPN with pre-trained ImageNet model, and train the RPN.

2) **Fast R-CNN training**: Train the detection network (Fast R-CNN) using the region proposals generated by step 1).

3) **RPN fine-tuning**: Fine-tune the layers which are unique to RPN. Fix the shared convolutional layers.

4) **Fast R-CNN fine-tuning**: Fine-tune the layers which are unique to Fast R-CNN. Fix the shared convolutional layers.

In the RPN training, intersection over union (IoU) scores between ground truth bounding boxes and proposed bounding boxes are used to generate positive (object) and negative (background) examples. The boxes with IoU score greater than 0.7 are treated as positive examples and the boxes of IoU score lower than 0.3 are treated as negative examples. The loss function for the RPN consists of a classification (of objectness) term and a box regression term. The trained RPN is utilized when Fast R-CNN is being trained. RPN generates the region proposals during the forward pass and weights of the RPN are also updated during backward pass of the Fast R-CNN. In this work, we use the pre-trained ImageNet model for Zeiler and Fergus network (ZFNet) [10] architecture to initialize the weights of step 1) and 2).

We aim to train a model for hand detection in this work. However, this work can be generalized and applied to other objects which are used in the stroke rehabilitation system such as torso, cylindrical object, hour-glass object, round-top object, and etc. The only issue for other object detection is how to collect sufficient amount of data for training. We do not deal with the data collection issue in this report because it is out of scope.

4 Experiments

4.1 Dataset and implementation details

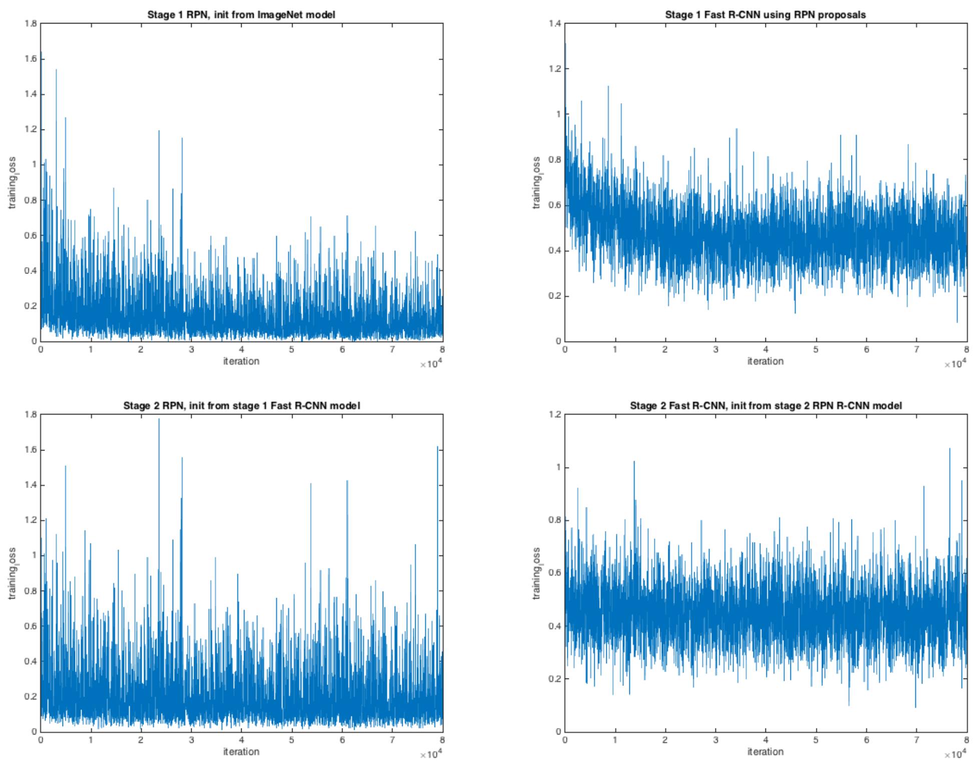

Figure 1: Training loss with respect to iterations. Each plot corresponds to each step of 4-step training pipeline.

We used Oxford hand dataset [8]. This dataset consists of 13050 hand instances: 9163 instances are used for the training and 2031 instances are used for the test. We did not use the validation instances. We used the python and C++ implementation of Faster R-CNN by the author [4]. The code is implemented with an open sourced deep learning library Caffe [9]. Hyper parameters such as the number of scales and aspect ratios of anchor boxes were set to the same values used in [4]. For the training and testing, we ran the system on a Tesla K80 GPU equipped linux machine. We used 80k iterations per each training stage and the entire training time took about 12 hours to train the model with the Oxford hand dataset.

We had a training loss convergence issue as shown in the Fig. 1. Training loss of RPN and Fast R-CNN decreased in stage 1 even though there were some amount of fluctuations. However, in stage 2, the training losses were not converged. We could not fully address this convergence issue. However it turns out that the trained model is working. We will further investigate this issue as a future work.

4.2 Quantitative results

The system achieved a mAP of 0.564 with ZFNet architecture [10]. If we use more advanced network architectures such as VGGNet [13] or ResNet [14], an increased mAP is expected. We would investigate this further as our future works.

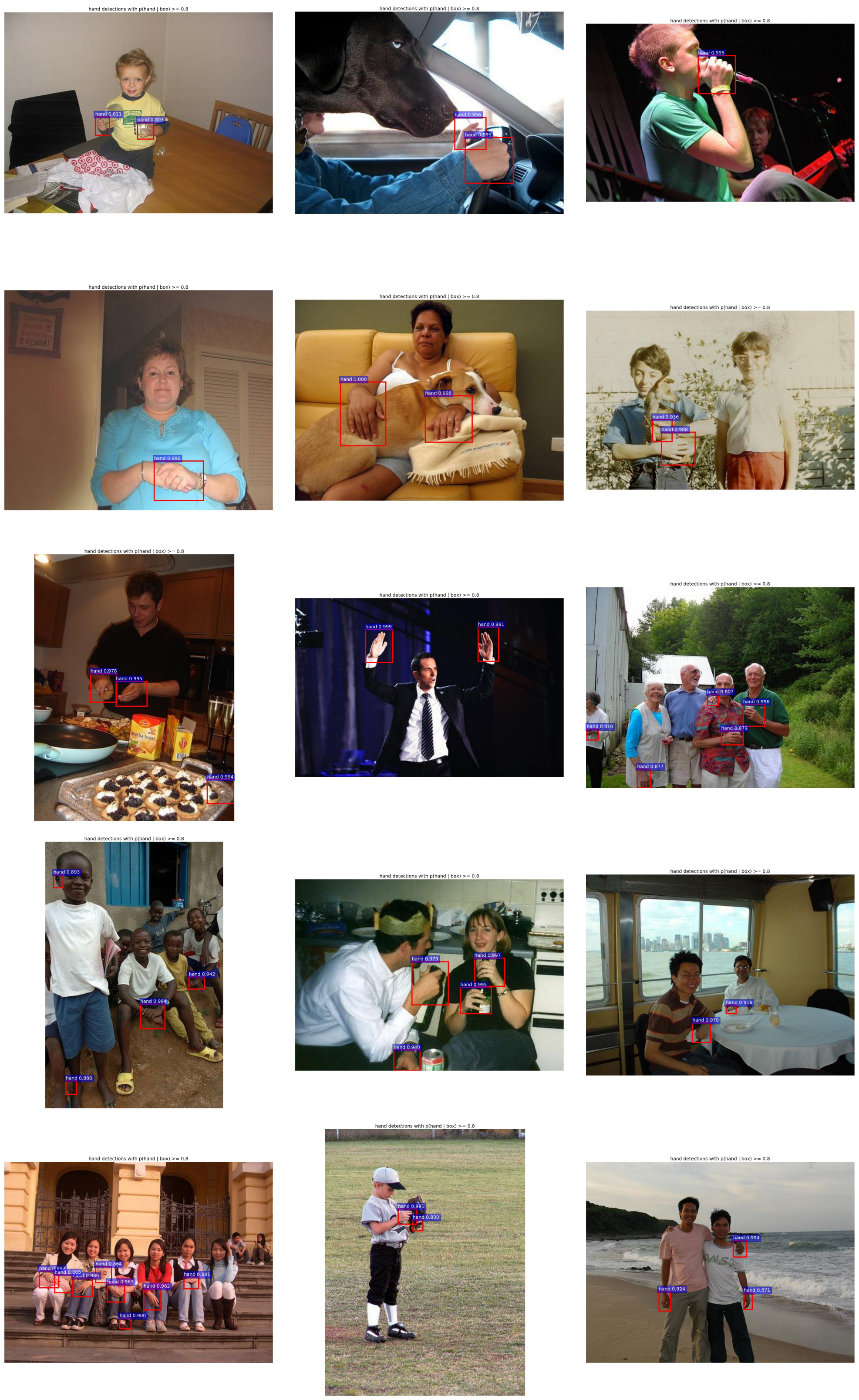

Figure 2: Successful cases of hand detection. Note that in spite of various illuminations, skin-tones, occlusions, poses the system detects the hands successfully.

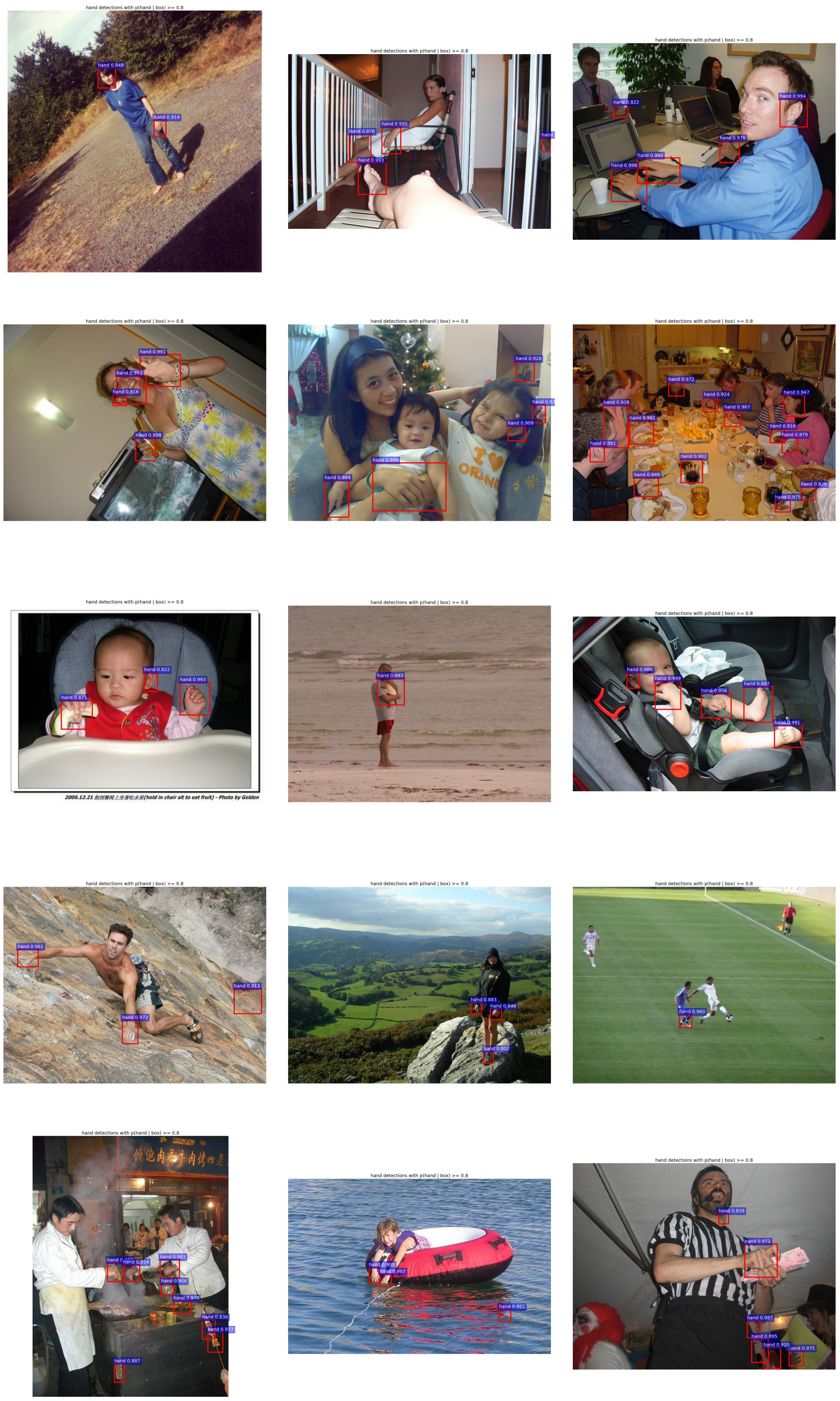

Figure 3: Failure cases of hand detection. Note that there are some false positives of similar textured objects such as ears and faces.

4.3 Qualitative results

Some successful detection results are depicted in the Fig. 2. We marked the detected hands with red bounding boxes along with the confidence values. Multiple hands with different skin-tones, various illumination conditions, small resolutions and severe occlusions are detected precisely. Some representative failure cases are illustrated in the Fig. 3. The system detects the hand when there are visible hands in the image. However, there are many false positive detections when the similar textured regions such as ears and faces exist. Incorporating negative images without any hands to the training dataset could be a possible solution to this problem. We expect that this would also increase the mAP score.

4.4 Test-time speed

Test-time speed of the hand detection is 100ms per image (=10 frames per second) on the average when using ZFNet [10] architecture on the linux machine equipped with Tesla K80 GPU. This is a reasonable speed for real-time detection module. However, if the rehabilitation system consist of more modules such as torso detector, grasp classifiers, and etc, 10fps hand detection would not be a sufficient speed. More speed up could be achieved if network compression techniques are employed.

5 Conclusion

We addressed the real-time hand detection problem for stroke rehabilitation system. We applied Faster R-CNN to get robust and accurate detection for various skin-tones and illuminations. The experimental results in this work are preliminary for the future steps. We will investigate deeper and advanced network architectures such as VGGNet [13] and ResNet [14]. Furthermore, we can apply CNNs to grasp recognition task and other detections task as well. The test-time computation complexity is still not very light, because for the stroke rehabilitation system, we have a grasp recognition and a tracking modules as well as a hand detection module. Thus, as another future work, we could apply the deep compression techniques [11], [12] to the system to get some further speed up.

Acknowledgments

This work is done with a supervision of Dr. Dhruv Batra, Dr. Devi Parikh, and Dr. Aisling Kelliher of Virginia Tech. This work is also submitted to ECE6554: Advanced Computer Vision, Spring 2016, Virginia Tech as a class project.

References

[1] Cheng Li and Kris M. Kitani, Pixel-level Hand Detection in Ego-Centric Videos, in CVPR 2013.

[2] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in CVPR 2014.

[3] Ross Girshick, Fast R-CNN, in ICCV 2015.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, NIPS 2014.

[5] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, You only look once: unified real-time object detection, CVPR 2016.

[6] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, Gradient-based learning applied to document recognition. Proceedings of the IEEE, November, 1998.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, Imagenet classification with deep convolutional neural networks, NIPS 2012.

[8] Arpit Mittal, Andrew Zisserman, and Phillip H. S. Torr, Hand detection using multiple proposals, BMVC, 2011.

[9] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, Caffe: Convolutional Architecture for Fast Feature Embedding, arXiv preprint arXiv:1408.5093, 2014.

[10] Matthew D Zeiler, and Rob Fergus, Visualizing and understanding convolutional networks, ECCV 2014.

[11] Song Han, Huizi Mao, and William J. Dally, Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016.

[12] Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin, Compression of Deep Convolutional Neural Networks for Fast and Low Power Mobile Applications, arXiv preprint arXiv:1511.06530, 2015.

[13] Karen Simonyan and Andrew Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, Deep Residual Learning for Image Recognition, arXiv:1512.03385, 2015.