Egocentric RGB Hand Detection

2017/12/04

by Mark Chang
(mark.fc_chang@htc.com)

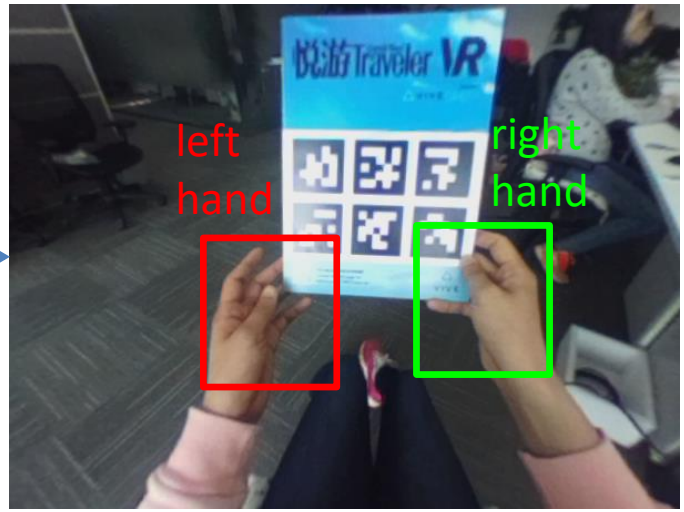
Outline

- Introduction
- Training a Egocentric RGB Hand Detector
- Rules of Competition
- Submission
- Evaluation Criteria
- Suggested Readings and Resources

Introduction

In this project, your work is to train a hand detector. The input of this hand detector is an RGB image containing one hand or two hands. Your hand detector should output the bounding-box of each hand, and the label indicating that it is left hand or right hand.

Hand Detector

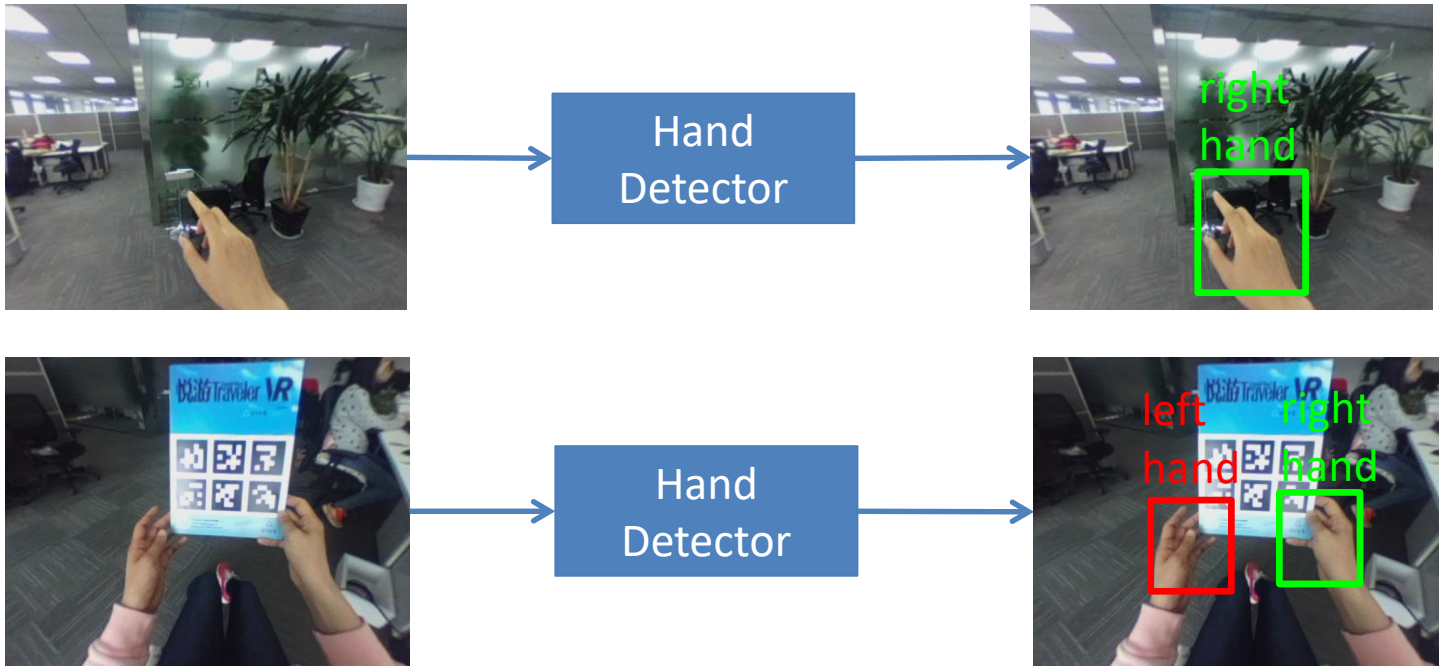


input:
Raw image

output:
bounding-box coordinates and class (left hand, right hand)

Introduction

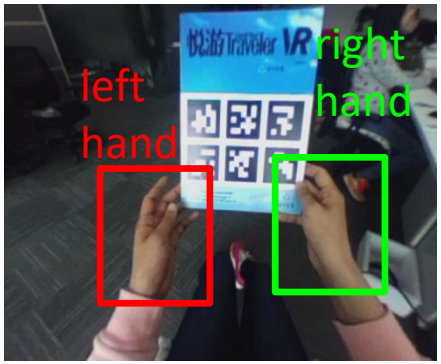
The input images may contain one hand or two hands. For each images, your hand detector should output the exact numbers of bounding-boxes and their corresponding labels.

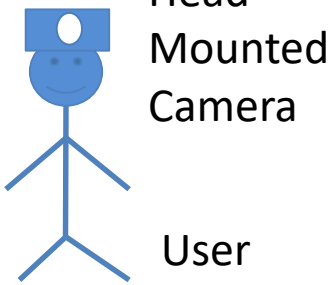


Egocentric View V.S 3rd Person View

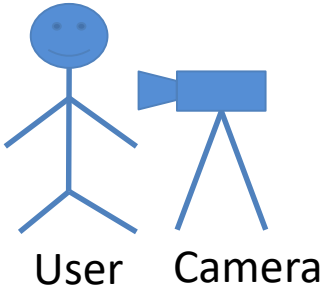
What is the difference between egocentric view and 3rd person view? In egocentric view, the user is wearing the camera so that the camera view is the same as the what user can see. However, in 3rd person view, the camera is beside the user or in front of the user.

1st person view (Egocentric view)





3rd person view



Egocentric Hand Detection

In this project, the hand detector should detect the hands of the person wearing the camera. However, the hands of other people should not be detected.

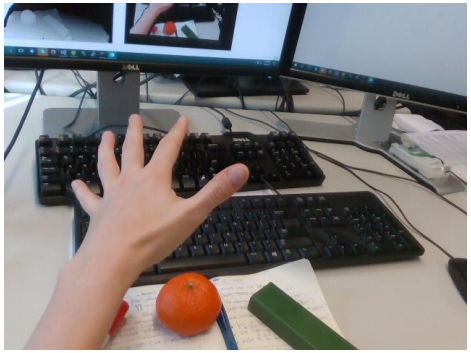
Hand of the user should be detected.

Hands of other person (or people) should not be detected.

RGB V.S RGB-D Data

What is the difference between RGB images and RGB-D images for the task of hand detection? The RGB images contain clutter background so RGB images is more challenging than RGB-D images for hand detection. However, RGB camera is more easily available. Hence, it has higher applicability.

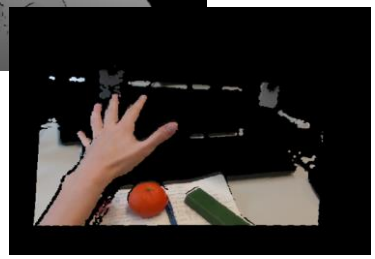
RGB Image



- Pros : High applicability
- Cons : More challenging for hand detection

Depth Image

RGB-D Image



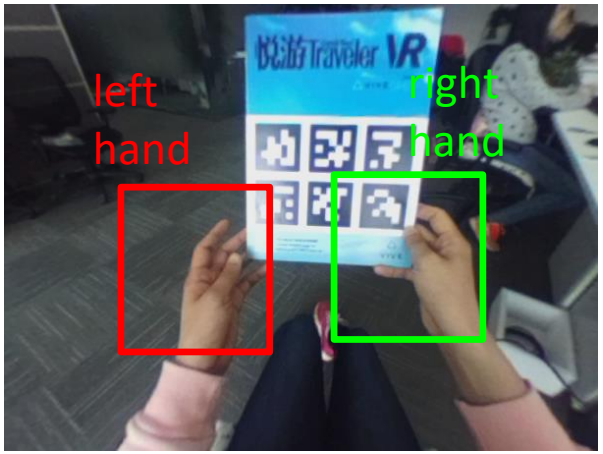
- Pros : Low applicability
- Cons : Less challenging for hand detection

photo credit :
http://resources.mpi-inf.mpg.de/handtracker/data/EgoDexter/data/Desk/

Why Using Synthetic Data?

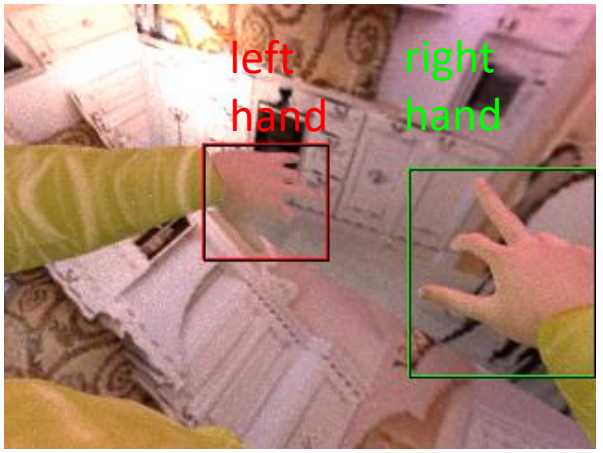
In many applications, the number of real images is usually not enough to train a deep learning model. Since synthetic images can be easily generated, we use synthetic images to increase the size of training dataset.

Real Images



- Pros : Training data is much similar to testing data.
- Cons : Creating a large-size dataset size is hard.

Synthetic Images



- Pros : Creating a large-size dataset is easy.
- Cons : Training data is not similar to testing data.

Training a Egocentric RGB Hand Detector

In this project, we provide lots of labeled synthetic images and some real images Your hand detector is trained on these images, and tested on real images.

Train the model by real & synthetic images simultaneously.

Test the model by real images.

Training Data

The training dataset contains synthetic images and real images. The synthetic dataset is the DeepQ-Synth-Hand dataset, and the real dataset is DeepQ-Vivepaper dataset.

DeepQ-Synth-Hand Dataset
10,0000 labeled images

DeepQ-VivePaper Dataset
464 labeled images
6197 unlabeled images

Testing Data

The testing dataset contains only real images from DeepQ-Vivepaper dataset. We will not release these images until the end of this competition. However, you can running your model on these images by submitting your model to our website**.**

hand-with-book images

hand-in-the-air images

DeepQ-VivePaper
504 labeled images for testing
229 hand-with-book images
275 hand-in-the-air images

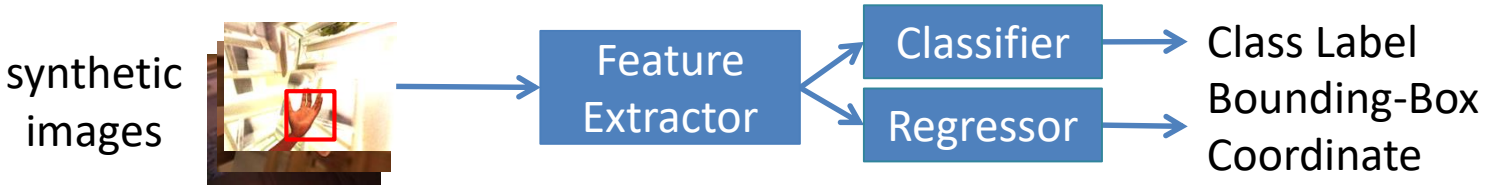
Methods for Training from Synthetic Images and real images

- There are some methods to train the object detector by synthetic images and real images.
- If the quality of synthetic images is good enough, we can directly train on synthetic images (without real images)
  - **1. Transfer Learning from ImageNet to Synthetic image**
- However, in this project, the quality of synthetic images is not good enough. To enhance the performance, we suggest you to add the real images into training dataset.
  - **2. Transfer Learning from ImageNet to Synthetic and Real image**
- If the training dataset contains real images, there are some advanced methods to improve the performance:
  - **3. Transfer Learning from Synthetic images to Real images**
  - **4. Domain Adaptation in Image Space**
  - **5. Domain Adaptation in Feature Space**
- You can choose any methods as you want, and you are welcome to invent your new method for this project.

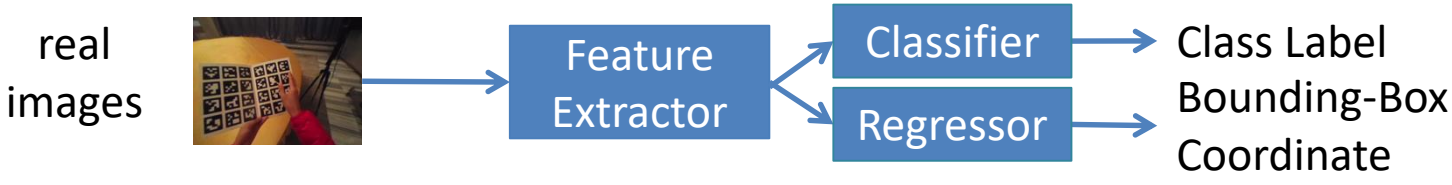
Transfer Learning from ImageNet to Synthetic image

1. Pre-train the model by imageNet.

2. Train the model by real & synthetic images simultaneously.



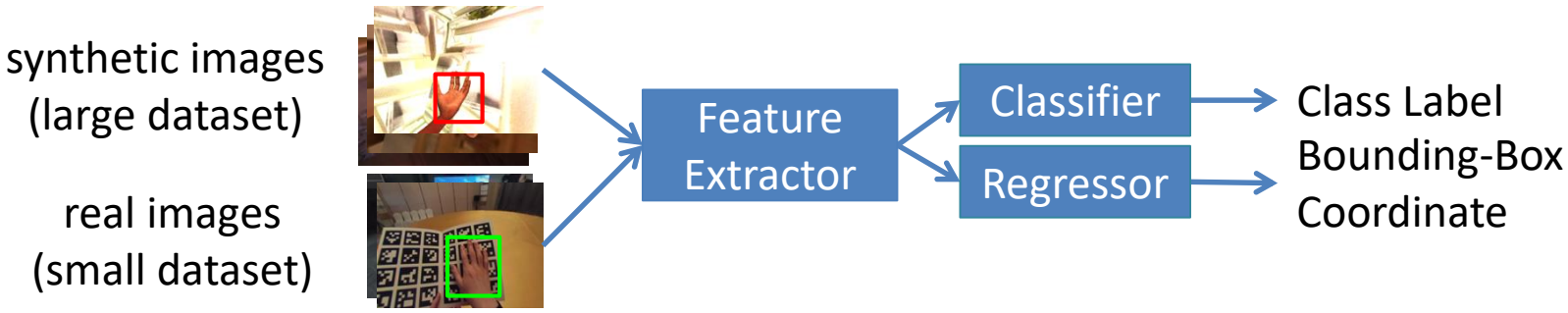
3. Test the model by real images.



Transfer Learning from ImageNet to Synthetic and Real image

In this project, the synthetic images are different from the real images. If you only train your model on synthetic images, the performance may not be good. Hence, we suggest you add some real images into your training dataset. The simplest method is train the model on both real and synthetic images simultaneously.

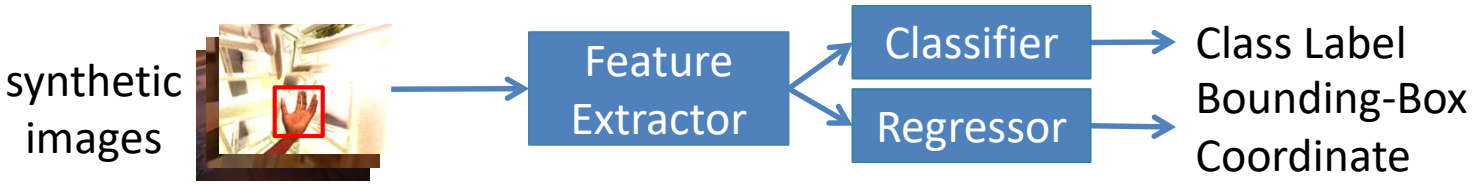
Pre-train the model by imageNet and train the model by real & synthetic images simultaneously.



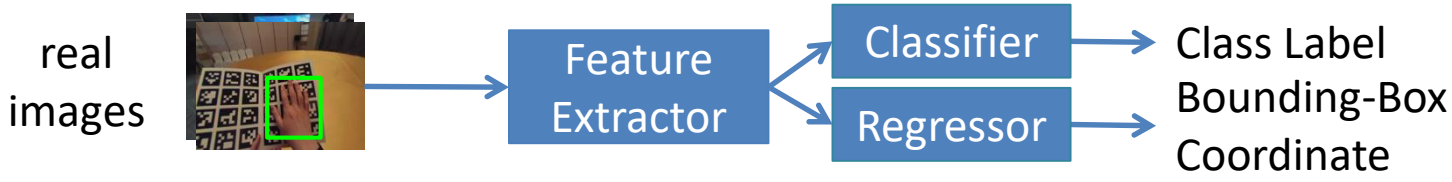
Transfer Learning from Synthetic images to Real images

However, there are too many synthetic images and too few real images. If you mix them together. The performance on real image may not be good. A better solution is to train the model by synthetic image first, and then fine tune the model by real images.

1. Pre-train the model by imageNet and Train the model by synthetic images.



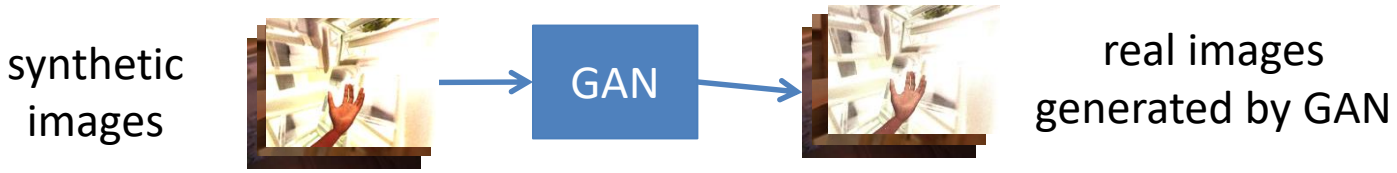
2. Fine-tune the model by real images.



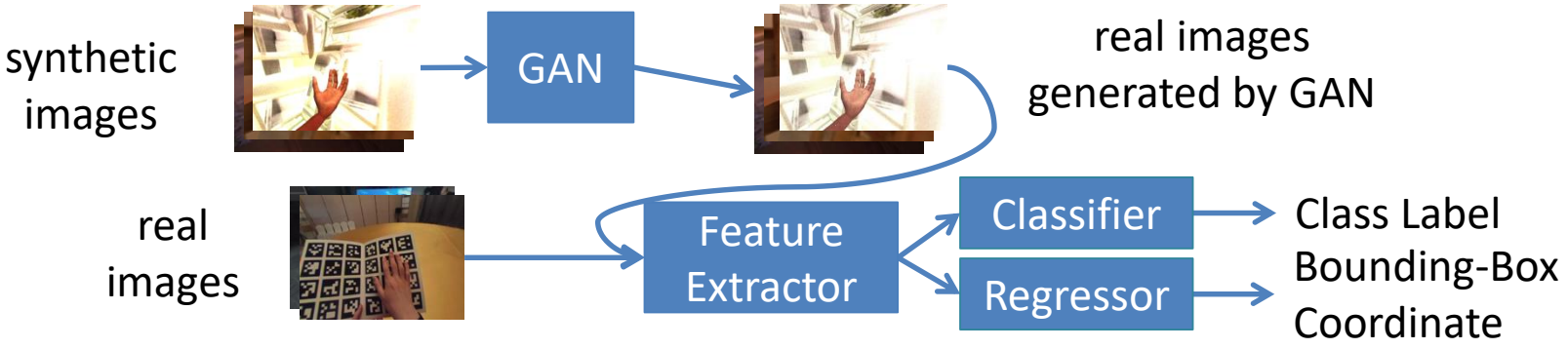
Domain Adaptation in Feature Space

Also, it is possible to “produce” more real images by synthetic images. If you know about generative model, you can apply conditional GAN to transfer the synthetic images into real images.

1. Train a conditional GAN to make the synthetic images look “real” .

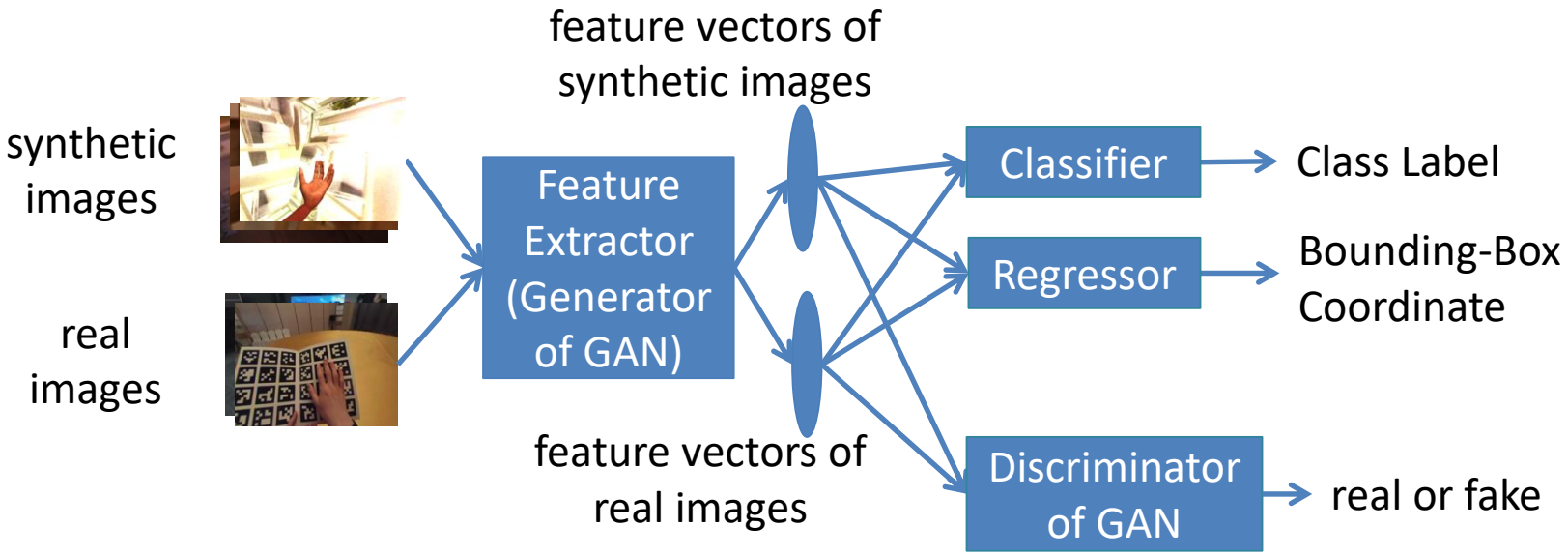


2. Transfer all the synthetic images into real images by conditional GAN, and train the model by all the images.



Domain Adaptation in feature space

GAN can also be applied in the feature space. It can increasing the similarity between the feature vectors of synthetic images and the feature vectors of real images.



Rules of Competition

- Pre-trained Model
- Training Data
- Code of Your Implementation

Pre-trained Model

- The pre-trained model should **only be trained by the ImageNet dataset for classification.**
- The pre-trained model should not be trained by the ImageNet dataset for object detection, or other tasks of ImageNet Challenge, or trained by other datasets such as Pascal-VOC or MS-COCO.
- You are allowed to download any available pre-trained model from the internet, but you should make sure that it is only trained by ImageNet dataset for classification.
- You are allowed to create your own pre-trained model by yourself from scratch, but it should be trained from randomly-initialized parameters.

Training Data

- We provide you both labeled data and unlabeled data.
- Data Augmentation and Data Cleaning are allowed, but they should be run by script.
- Manually creating new data from existing data or manually deleting existing data are not allowed.
- Manually collect or create your own dataset are not allowed.
- The rules of using labeled data and unlabeled data are given in the following two slides.

For Labeled Data

- You are only allowed to use the labeled data in the given datasets (DEEPQ-Synth-Hand, DeepQ-Vivepaper) in addition to “ImageNet dataset for classification task”.
- You are allowed to use the label of keypoints and masks in DeepQ-Synth-Hand Dataset for multitask learning.
- Do not use any other labeled dataset (such as Pascal VOC, MS-coco, etc)
- Automatically create new labels or automatically adjusting the existing labels by running scripts are allowed.
- Manually creating additional labels or manually adjusting the existing labels are not allowed.

For Unlabeld Data

- You are allowed to use the unlabeled data in the given dataset (DeepQ-Vivepaper).
- You are allowed to use any other dataset available from the internet as unlabeled data. If the dataset contains label, you should assume that it is unlabeled.
- For domain adaptation algorithm, you are allowed to create the domain label (“real” or “synthetic”) for adversarial training.

Code of Your Implementation

- Your algorithm should be implemented in python programming language.
- Your algorithm should be implemented in “tensorflow”, “pytorch”, “keras”, “CNTK” or “mxnet” packages.
- Any other programming language or any other deep learning packages are not allowed.
- You are allowed to reuse any code available from the internet. However, you should be aware of its license.

Submission

- Submission for evaluating on testing data
  - During this competition, you can submit your model to be evaluated on testing data.
- Final submission
  - After you complete your project, you should submit your whole project so that we can verify your result.

Submission for Evaluating on Testing Data

- You must submit your “model” and “code to run your model” to **our submission site** for evaluating on testing data.
- Your code must include **our python package**, so that it can get the testing data and we can evaluate your result.
- There are three functions in our package:
  - get_file_names()
    Takes no argument, will return a list of file paths of the testing images.
  - get_output_file_object()
    Takes no argument, will return a file object for you to write your result.
  - judge()
    Takes no argument, will return a floating point number indicating your accuracy and a string to indicate the error encountered if any.
    accuracy, err = judge()
- **You can see your own accuracy on your testing data**, but we will not provide ranking information.

Submission for Evaluating on Testing Data

The code to run your model consists of the followings:

1. Call get_file_names() to get the list of files.
2. Load your model.
3. For each image, inference the answer.
4. Call get_output_file_object() to get the output file object.
5. Write the output.
6. Call the judge() to get the result.

Note that the timestamp created when judge() is called is used for judging if the submission is on time.

Submission for Evaluating on Testing Data

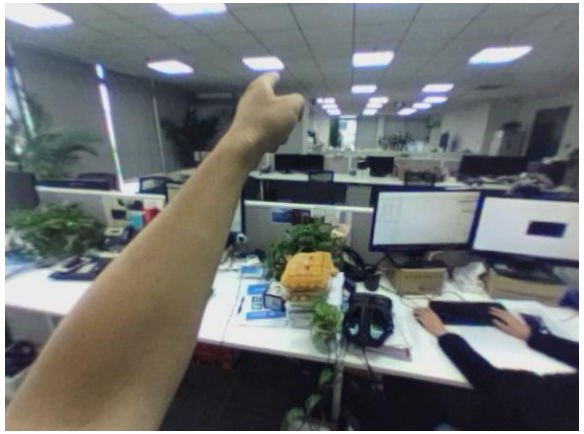
- Submission will take plenty of time, especially if your model is large. The logged time in our accuracy log (which will be produced in our scoring function) will be used to judge if the submission is on time.
- There will be no limit on the number of submissions made per team. But our submission site will only allow each team to run a total of 10 hours.
- The submission deadline, the submission website and a python package for debug use will be announced later.

Return of get_file_names()

After you calling the get_file_names(), it will return a list of file paths of the testing images. For example, if the testing set contains two images: “/data/image1.png” and “/data/image2.png”, this function will return:

```
[“/data/image1.png”,”/data/image2.png”]
```

/data/image1.png



/data/image2.png

get_output_file_object()

Your code should write the result to the file object. The format is at the following:

- image_path : the paths of images return from get_file_names()
- x0, y0, x1,y1 : four integers representing the coordinates of the bounding box.
- left_or_right : an integer between 0 and 1. Left hand is 0, and right hand is 1.
- score : a floating number between 0~1, for calculating mean average precision.

Notice that (1) Each line represents a distinct bounding-box, different bounding boxes of a given image should be written in two separated lines. (2) The space between each token is white space, not tab.

```
image_path1 x0 y0 x1 y1 left_or_right score
image_path1 x0 y0 x1 y1 left_or_right score
image_path2 x0 y0 x1 y1 left_or_right score
image_path2 x0 y0 x1 y1 left_or_right score
image_path2 x0 y0 x1 y1 left_or_right score
…
```

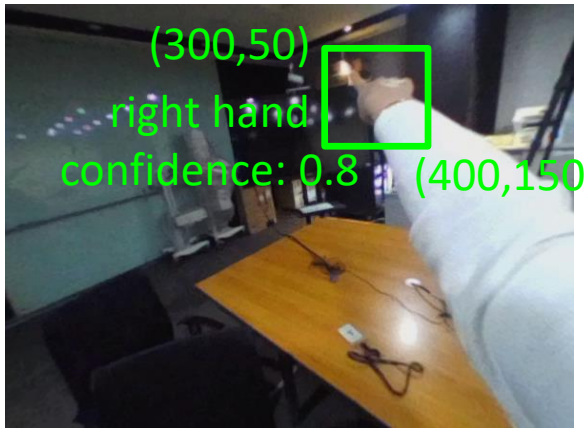
get_output_file_object()

For example, if your results are the same as the right images, your code should write the following lines to the output file object:

```
/data/image1.png 200 50 300 150 0 0.9
/data/image1.png 400 250 500 350 0 0.5
/data/image2.png 300 50 400 150 1 0.8
```

/data/image1.png

/data/image2.png



Final Submission

- Your final submission should contain the following:
  - Trained model & Whole Code
  - Document & report

Trained Model & Whole Code

- You should submit your trained model of your final result as well as the related code that can inference on this model.
- Besides the code mentioned above, you should submit all the code that can reproduce your final result from scratch. **If we fail to reproduce, we may consider you cheating, and you may be disqualified in this competition.**
- You should not submit your pre-trained model or dataset. Your dataset should be available from the internet.
- If your pre-trained model is created by yourself, your code should be able to train your pre-trained model from randomly-initialized parameters. Otherwise, your pre-trained model should be available from the internet.

Format of Document

- You should submit a document of your implementation.
- The document is about **how to run your code and reproduce your result. If we follow your document, your final result should be able to reproduce.**
- This document should contain the following:
  - Where to download your training data and pre-trained model
  - How to set up the environment for your code.
  - What is your final result, and how to run your code to reproduce your result.

Format of Report

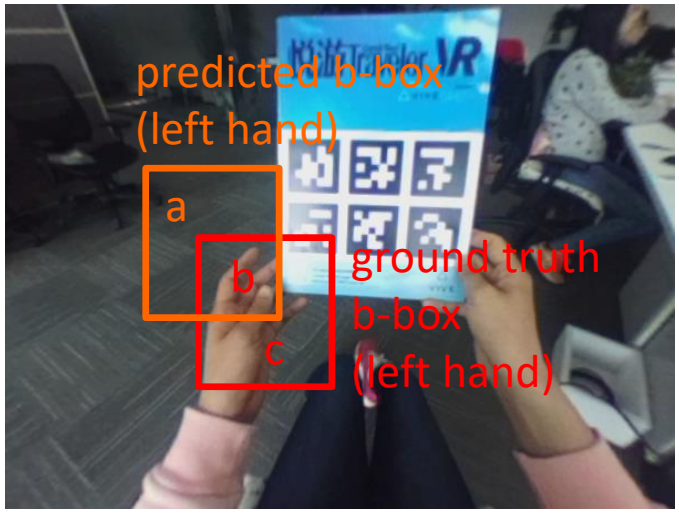
- Besides document, you should submit a report to introduce the theoretical and experimental parts of your project.
- Your report should be written in one-column or two-column conference paper style.
- Your report should contain at least (but not restricted to) the following sections:
  - Introduction
  - Proposed Method
  - Experiment
  - Conclusion

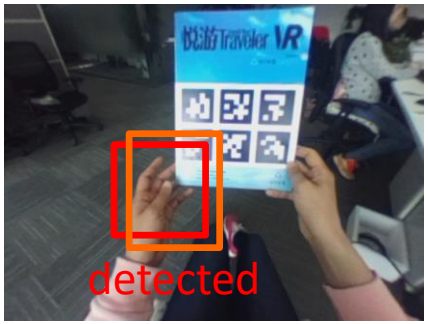
Evaluation Criteria

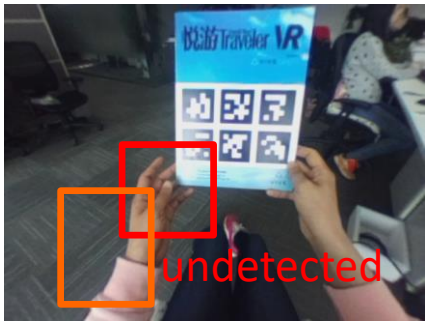
- 60% : accuracy
  - mean average precision (mAP)
  - 0.5 * (mAP of hand-with-book images) + 0.5 * (mAP of hand-with-book images
- 0 % : Model Size
  - Model size is not considered in our evaluation, However, your model should not be larger than **500 MB**. Whose model larger than this size will be disqualified in this competition.
  - You can use any data-compression method to reduce your model size. However, the compression algorithm must be run by your script automatically, and the decompressed model should be deleted automatically, too.
- 0 % : Runtime
  - Runtime is not considered, too. However, our submission site will only allow each team to run a total of 10 hours.
- 40% : Document, Report & Oral Presentation

Detected Bounding Box

- Detected: >=50% **Intersection Over Union (IOU)** between predicted bounding box(b-box) and ground truth b-box
  - Intersection : b
  - Union : a + b + c
  - IOU : b/(a + b + c)
  - Detected : IOU >= 50%
  - Undetected : IOU < 50%

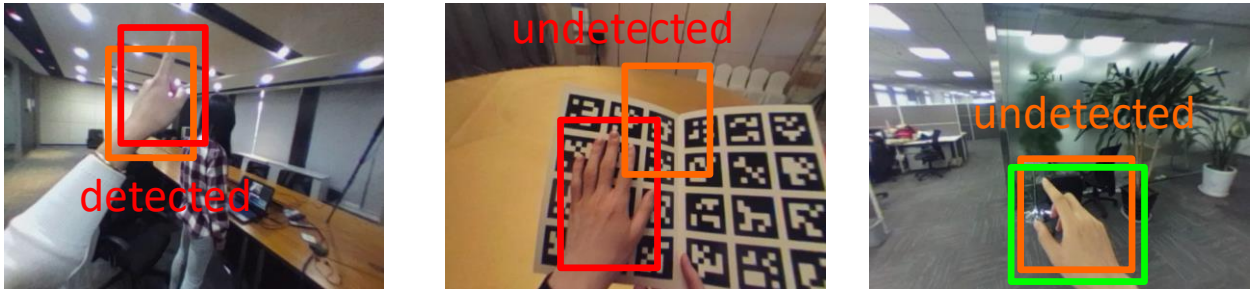






Precision & Recall

- Precision : (Nums of detected b-box) / (Nums of predicted b-box)
- Recall : (Nums of detected b-box) / (Nums of ground-truth b-box)



orange: predicted b-box of left hand
red: ground-truth b-box of left hand
green : ground-truth b-box of right hand
In this example, precision = 1/3, recall = 1/2, F-score:2/5

Mean Average Precision

- Average Precision (AP) : Average the precisions from low recall to high recall.
- Mean Average precision (mAP) :
  - average the APs over all classes

  (left hand and right hand)

score	precision	recall	max precision	average precision
>=0.9	1.0	0.33	1.0	0.7567
>=0.8	0.5	0.33	1.0	
>=0.7	0.67	0.67	0.67	
>=0.6	0.5	0.67	0.67	
>=0.5	0.6	1.0	0.6	

Suggested Readings and Resources

- Suggested Readings for Object Detection or Hand Detection
- Suggested Readings for Training by Synthetic Data
- Suggested Source Code for Object Detection
- Suggested Unlabeled Dataset for Adversarial Training

Suggested Readings for Object Detection or Hand Detection

- Faster R-CNN
  - https://arxiv.org/abs/1506.01497
- Mask R-CNN
  - https://arxiv.org/abs/1703.06870
- YOLO2
  - https://arxiv.org/abs/1612.08242
- Lending A Hand: Detecting Hands and Recognizing
  - http://vision.soic.indiana.edu/papers/egohands2015iccv.pdf
- Robust Hand Detection and Classification in Vehicles and in the Wild
  - http://adas.cvc.uab.es/cvvt2017/wp-content/uploads/sites/14/2014/03/5.pdf

Suggested Readings for Training by Synthetic Data

- Transfer Learning
  - https://arxiv.org/abs/1310.1531
  - https://arxiv.org/abs/1701.01370
- Domain Adaptation in Image Space
  - https://arxiv.org/abs/1611.01331
  - https://arxiv.org/abs/1612.07828
  - https://arxiv.org/abs/1703.10593
- Domain Adaptation in Feature Space
  - https://arxiv.org/abs/1409.7495
  - https://arxiv.org/abs/1604.02703
  - https://arxiv.org/abs/1702.05464

Suggested Source Code for Object Detection

- Faster-RCNN (tensorflow)
  - https://github.com/endernewton/tf-faster-rcnn
- Faster-RCNN (pytorch)
  - https://github.com/ruotianluo/pytorch-faster-rcnn
- Mask-RCNN (MxNet)
  - https://github.com/TuSimple/mx-maskrcnn
- Deformable-ConvNets (MxNet)
  - https://github.com/msracver/Deformable-ConvNets
- Darkflow (tensorflow)
  - https://github.com/thtrieu/darkflow

Suggested Unlabeled Dataset for Adversarial Training

- SCUT-Ego-Finger Dataset (Ego, Real Images)
  - http://www.hcii-lab.net/data/SCUTEgoFinger/index.htm
- Egohand Dataset (Ego, Real Images)
  - http://vision.soic.indiana.edu/projects/egohands/
- GRASP UNDERSTANDING DATASET (Ego, Real Images)http://www.gregrogez.net/research/egovision4health/gun-71/
- CMU-EDSH Dataset (Ego, Real Images)
  - http://www.cs.cmu.edu/~kkitani/datasets/
- EgoDexter Dataset (Ego, Real Images)
  - http://handtracker.mpi-inf.mpg.de/projects/OccludedHands/EgoDexter.htm
- SynthHands Dataset (Ego, Synthetic Images)
  - http://handtracker.mpi-inf.mpg.de/projects/OccludedHands/SynthHands.htm
- VIVA Dataset (Ego & 3rd , Real Images)
  - http://cvrr.ucsd.edu/vivachallenge/
- Oxford Hand Dataset (3rd, Real Images)
  - http://www.robots.ox.ac.uk/~vgg/data/hands/
- Hand3D Dataset (3rd, Synthetic Images)
  - https://lmb.informatik.uni-freiburg.de/projects/hand3d/

If you have any question about the rules, the dataset, or any other question about this competition, please send an email to Mark Chang mark.fc_chang@htc.com , thanks.