Image-based Object Detection System for Self-driving Cars Application

The scope:

Self-driving car is the exciting area, which can be interpret a world-class Artificial Intelligence task. It includes Machine Learning and Control Theory.

In Machine Learning, building a self-driving car needs the engineers handles: Fundamentals of Machine Learning (Classification and Regression), Different Neural Networks, Logistic Classifier, Optimization, Transfer Learning, Sensor Fusion (Bayesian Related), Kalman Filters, Particle Filter, Markov Localization.

Deep Learning provide a cheaper and more reliable way for car to understand the scene better by giving better objects detectors and trackers.

In this project, we are targeting to build an image-based state-of-art object detector from scratch from a given driving data.

Let us have a COMPETITION.

Project Setting:

Goal: Detect 4 different kinds of objects in given images by training 10k annotated training images. The detection precision is measured by drawing bounding box around the objects. Four different objects includes vehicle, pedestrian, cyclist and traffic lights (labeled as 1, 2, 3, 20).

As a team: Up to 3 people per team. It is also encouraged to do this project by only one person.

Report: Every team needs submits the Model, Code and Documentation/Presentation Slide for final evaluation.

Deadline: September 20th

The Data

The data is provided by Bittiger. Please do not share or spread the data without permission from Bittiger.

There are 10000 well-annotated images as training data. Each image has bounding boxes of 4 different objects. All the bounding boxes with class labels are in ".idl" files. Each team uses these 10000 images as training dataset and validation dataset as its will. There is also a testing dataset for testing which doesn't have ground truth bounding box information.

Training Dataset A: https://s3-us-west-2.amazonaws.com/us-office/competition/training.zip
Testing Dataset B: https://s3-us-west-2.amazonaws.com/us-office/competition/testing.zip

Detail Information:
Each image is 640 x 360
Training data annotation information is in "label.idl", each line of the annotation is a "json" string. For example:

{"60109.jpg": [[158.66624, 200.00016000000002, 182.1664, 218.16683999999998, 1], [0.16666688, 181.00007999999997, 99.00031999999999, 292.5, 1], [193.50016, 200.75004, 215.25056, 217.00008000000003, 1]]}

,which is a dictionary. The key is the name of the image, then along with a list of array. Each array represents a bounding box.

Each bounding bounding box has 4 digits:
［box_upper_left_x, box_upper_left_y, box_lower_right_x, box_lower_right_y, class_label]
Class label has 4 categories: 1, 2, 3, 20 (vehicle, pedestrian, cyclist and traffic lights)

The evaluation

The code, model, json file are in the team's github as submission.

The team run the detection algorithm on 2000 testing images with bounding box. The acceptable file format is "json string" like:

{ "image_filename": [[upper_left_x, upper_left_y, lower_right_x, lower_right_y, class_label, confidence], …], … }

Sample Result:

```
{

  "70690.jpg":[ [23.0,177.00012,155.00032,275.00004,1,0.895981],

    [162.99968,209.99988, 211.00032,245.99988,1,0.996691],

    [206.0,209.00016,236.0,235.00008,1,0.920521] ],

 "71927.jpg":[ [230.0,168.99983999999998,260.0,196.99992,1,0.973079 ],
    [181.00032,173.99988000000002, 208.99968,198.00000000000003,1, 0.946471 ],
    [207.00032,173.00016,224.99967999999998,191.00016000000002,1,0.0189044 ]  ]

}
```

The results is a dictionary, the key is the image name, the value is the list of bounding box. The four coordinates are x and y of left-top, right-bottom, class label, and the confidence (0~1). Please dump the results into json file.

The evaluation method is Average Precision score (Precision-Recall Curve).

Weekly Meeting Setting:

Each team decides to present their weekly work or not.
A team have 2 minutes to talk about their work if that is the case.
Each team talked about the difficulties/blocks at will during the meeting, and the project supervisor decides to discuss the issue online or not.
If the question is not discussed in the during, the issue will be answered by supervisor offline.
Supervisor is/isn't going to have a talk/teaching section after team discussion based on situation.

Preparation:

Hard Requirement:

A Github: each team should have a github which is place all the code, documentation and presentation slide (good for resume).

An AWS account OR Desktop with recent GPU: For AWS, each team needs to budget themselves up to 100 dollars for modeling training. The members in the team can share the cost. The team decides the details.

A project tracking documentation: record the team activity every week including plan and review.

Soft Requirement:

Team culture build: be honest, be responsible. It is highly recommended to have the daily/mid-week sync. Members should have a clear and dynamic picture. And most important, be fun.

Some useful guidance:

This capstone project can be divided into two main parts:
Paper investigation and Algorithm Implementation

1. To gain the full picture of state-of-art object detection algorithm, it is necessary to do a full investigation/exploration. A week or so should be sufficient, which is limited by our timeline.
2. To accelerate the progress of the project, the best way is standing on the giant's shoulder.
3. An paper exploration example can be:
    a. Start with survey paper (split the paper reading tasks to members, and sync the knowledge) . The survey paper should be read in details. Then jump into massive reference or interesting recent paper with key words. Filter majority of the paper by reading through the abstract and conclusion.
    b. Read selected paper in great details (5 paper at most)
    c. Pick the most interesting paper.
    d. Search its implementation in existing framework
4. An project timeline can be (should controlled by the team)

Week 1 : Paper Reading & Algorithm Decision & Framework decision & Data Investigation & Data format Convert

Week 2: Scripting, the details can take MNXET codelab as a possible example. However, there are a lot of open-source code and frameworks. The team is responsible to choose what they think is the most efficient way. Because our timeline is really tight.

Week 3:Training the network <-> Modify the network architecture. This is the most painful part, be prepared.

Week 4: Keep Training if any possible improvement. Write report and document everything.

Beyond: If the team complete the project so early, try to build the model on real time detection system. For example, the team can apply the model on video or real camera in car. And the potential issue the team may encounter with is the speed requirement.

5. Two things in mind: budget and timeline.

Some useful links:

Precision-Recall, Average Precision
Awesome Deep Learning Paper
Awesome Object Proposals
Awesome Deep Vision
Everybody loves Tensorflow
How to do a Deep Learning Project
Faster RCNN Caffe Implementation
Faster RCNN TF implementation
Faster RCNN MXNET implementation
Yolo official Web
Yolo Tensorflow
Yolo MXNET
R-FCN: the paper
R-FCN TF
R-FCN Caffe
R-FCN MXNET