Nexar deep learning challenge II

Vehicle Detection in the Wild using the NEXET Dataset

Rules & conditions:

- Include running code, and dependencies
- 5 vehicle categories: car, van, pickup-truck, truck, and bus.
- Includes images at night
- Open source license only!
- No hand-labelling :-(

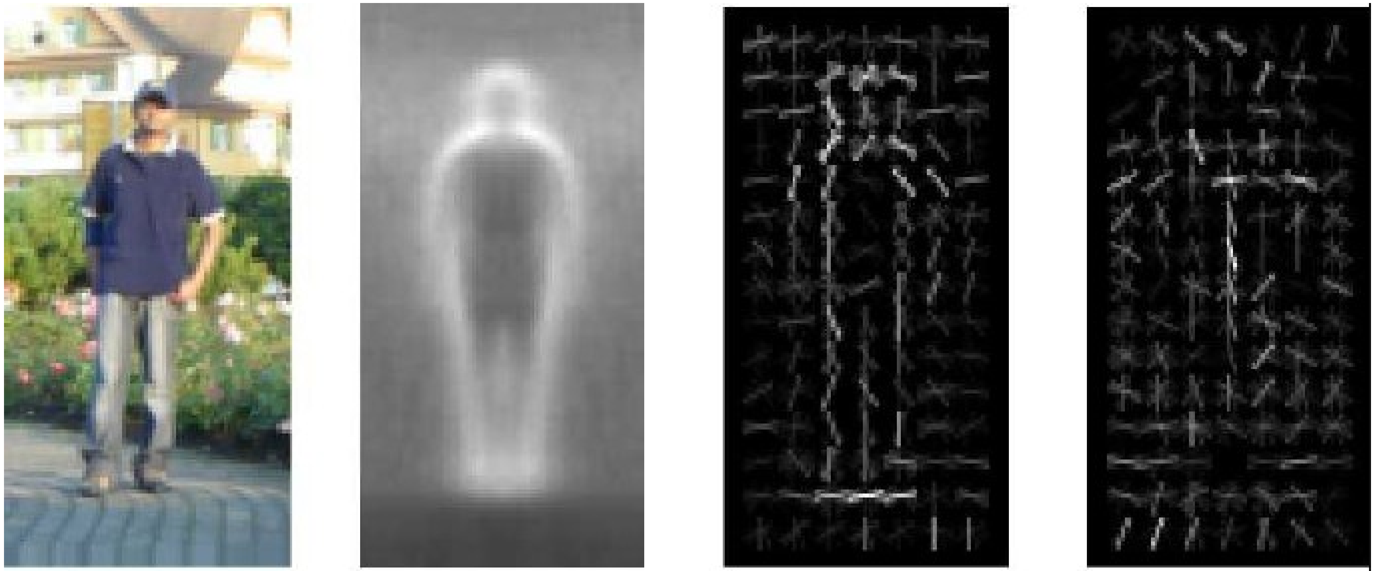
Object detection pitching idea...

Title	Approach	Code available
YOLO9000: Better,Faster,Stronger	'YOLO2'	https://pjreddie.com/darknet/yolo/ C+Cuda+config files
SSD: Single Shot MultiBox Detector	Discretized default bounding box shape	https://github.com/balancap/SSD-Tensorflow https://github.com/weiliu89/caffe/tree/ssd
Perceptual Generative Adversarial Networks for Small Object Detection	GAN for super-resolution	?
Reinforcement Learning for Visual Object Detection	RL / progressive fixation and evidence	?
Tree-Structured Reinforcement Learning for Sequential Object Localization	RL / Q-learning of progressive image 'crop'	?
Hierarchical Object Detection with Deep Reinforcement Learning	RL / Q learning of progressive image 'crop'	https://github.com/imatge-upc/detection-2016-nipsws
Faster R-CNN...	CNN	https://github.com/rbgirshick/py-faster-rcnn

Older methods, and base building blocks

Name	Link	N.B.
VGG-16	https://github.com/fchollet/keras/blob/master/keras/applications/vgg16.py	Building block for 'some' of the papers, esp. RL related
HOG / misc feature systems	http://www.vlfeat.org/api/hog.html	Combined with cascaded classifiers

(historical) Example of pre-filter: histogram of oriented gradient



Input example

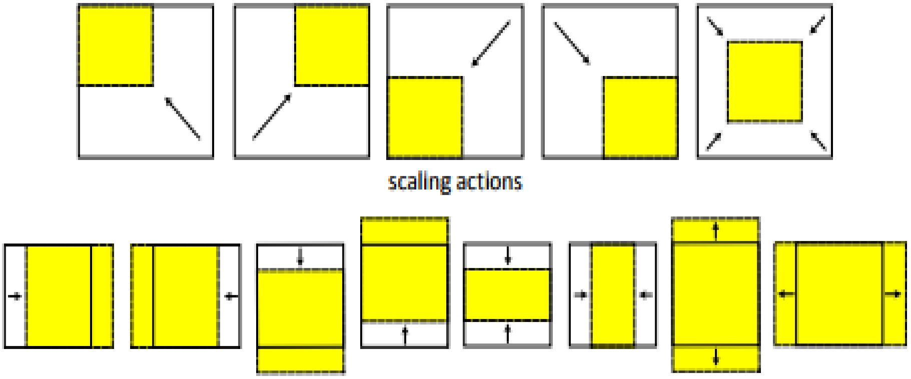
Average gradients

Weighted pos wts

Weighted neg wts

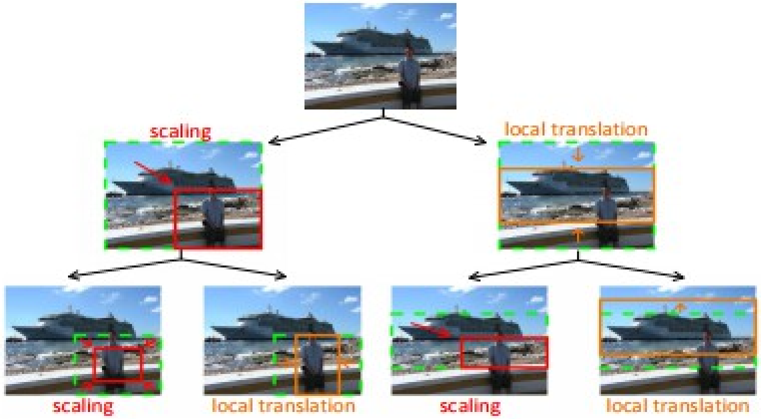
Many more, sometimes appear in fairly recent papers

RL/Trees [1]
13 alternatives per 'branch'

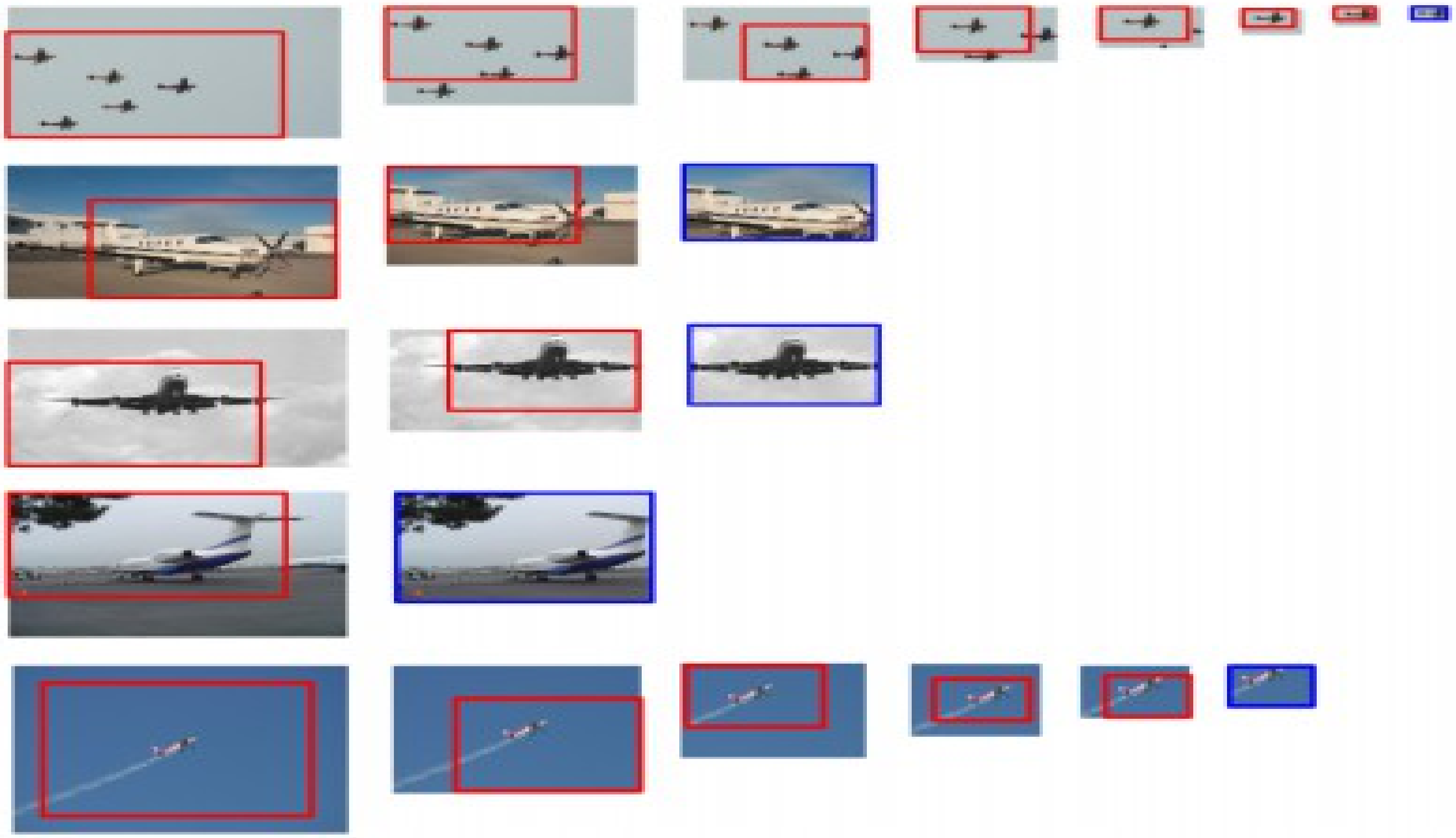


scaling actions

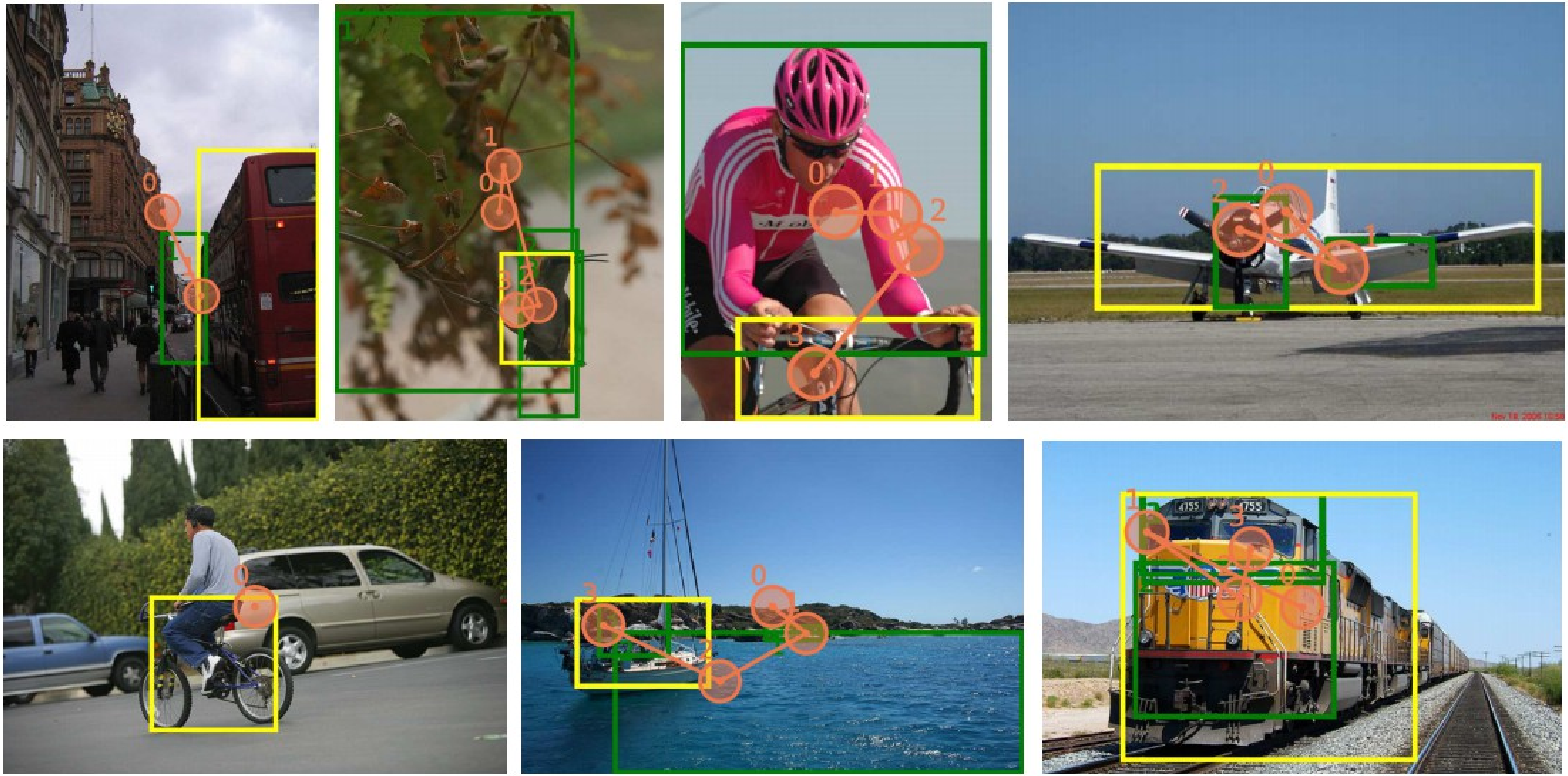
local translation actions



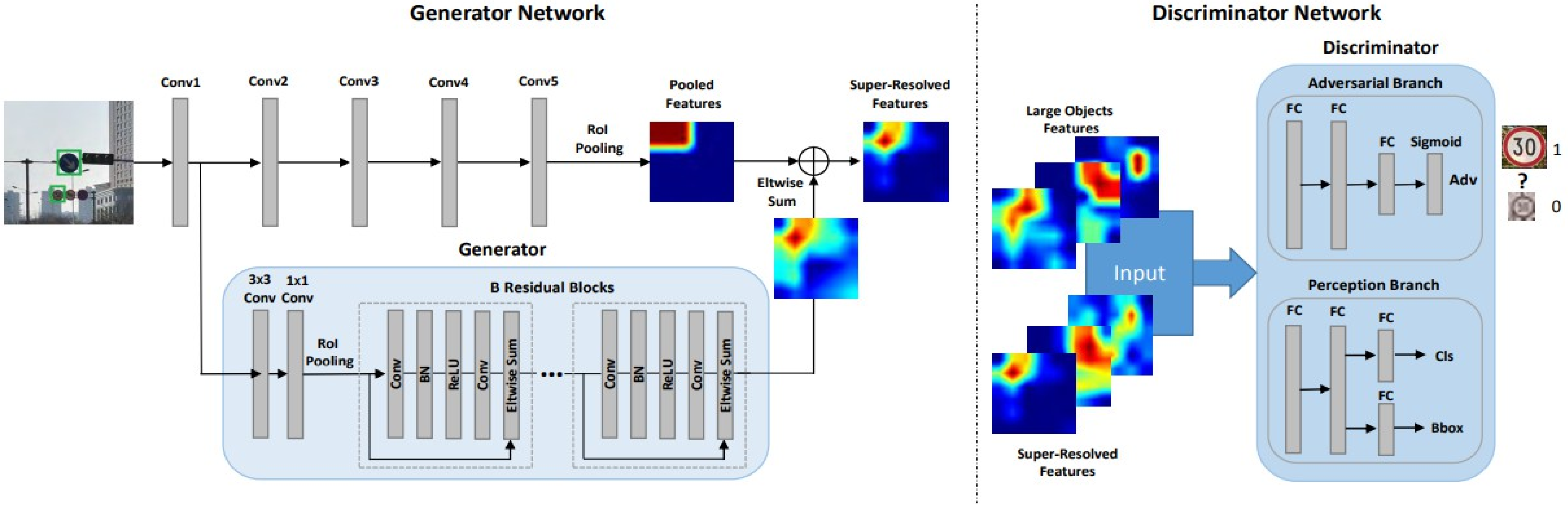
RL/Trees [2]
6 alternatives per 'branch'



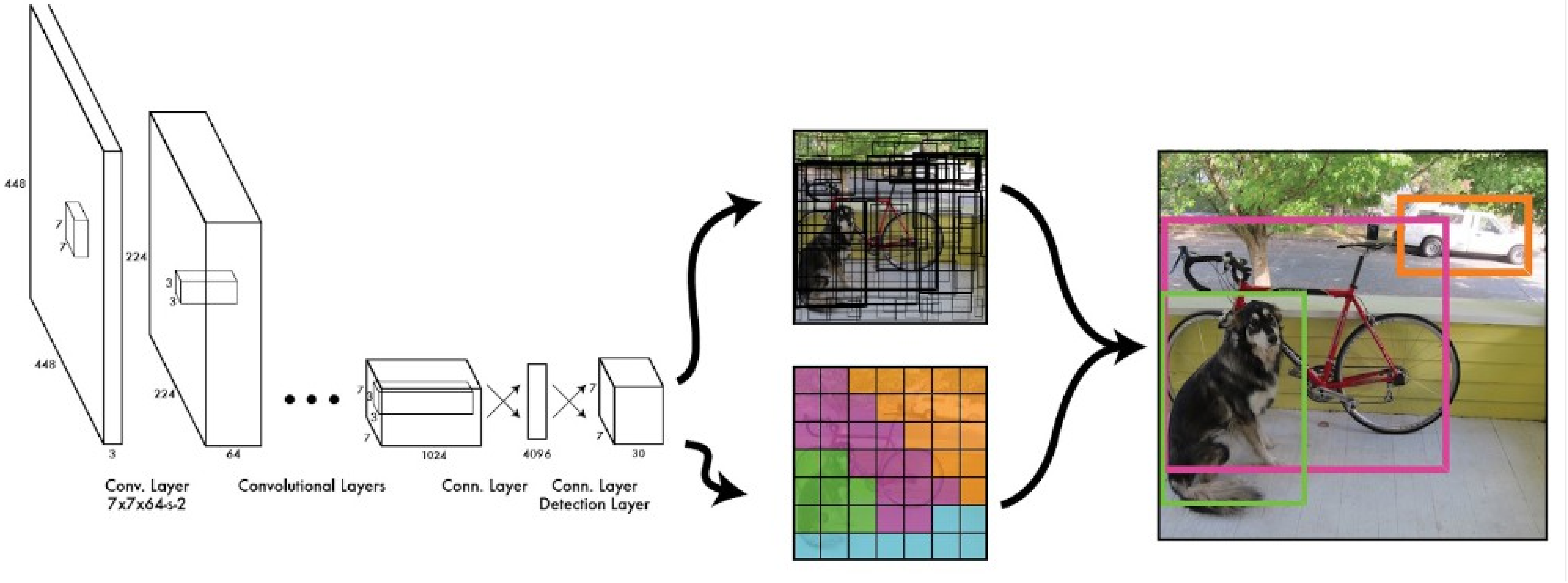
Fixation & evidence based illustration



Small object detection GAN



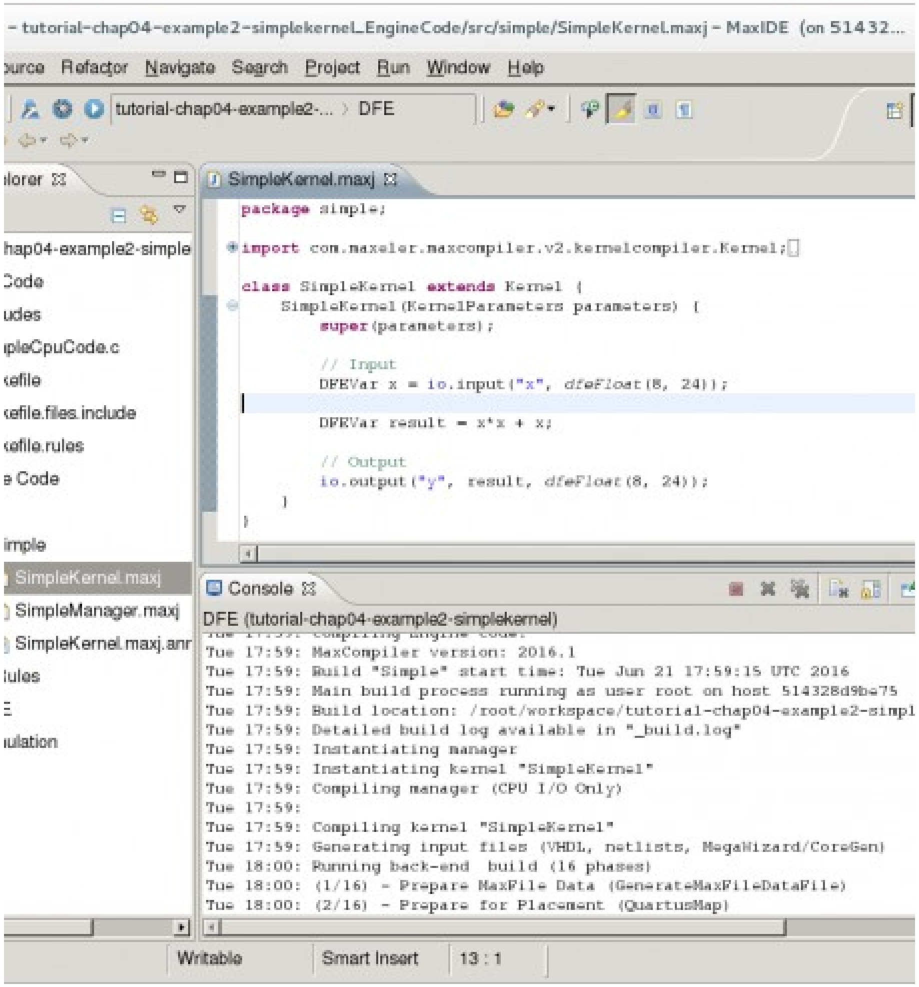
YOLO – You only look once



Several approaches
… need some help to choose wisely

- Bounding boxes
- Bounding boxes (RL)
- Features
- Detection GAN: hyperesolution + small objects ("macro-features" ?)
- How to improve robustness? (GAN?)

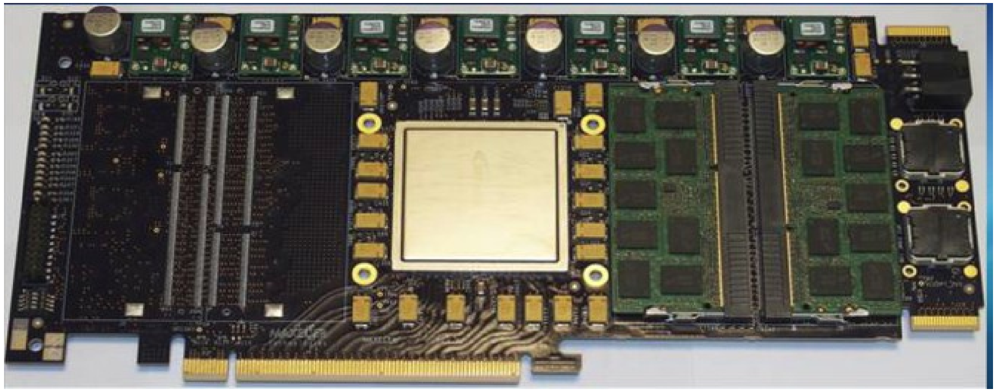
Embedded side: FPGA (one proposal, later phase)



```
package simple;

import com.maxeler.maxcompiler.v2.kernelcompiler.Kernel;

class SimpleKernel extends Kernel {
    SimpleKernel(KernelParameters parameters) {
        super(parameters);

        // Input
        DFEVar x = io.input("x", dfeFloat(8, 24));

        DFEVar result = x*x + x;

        // Output
        io.output("y", result, dfeFloat(8, 24));
    }
}
```

DFE (tutorial-chap04-example2-simplekernel)

```
Tue 17:59: Compiling Engine Code:
Tue 17:59: MaxCompiler version: 2016.1
Tue 17:59: Build "Simple" start time: Tue Jun 21 17:59:15 UTC 2016
Tue 17:59: Main build process running as user root on host 514328d9be75
Tue 17:59: Build location: /root/workspace/tutorial-chap04-example2-simpl
Tue 17:59: Detailed build log available in "_build.log"
Tue 17:59: Instantiating manager
Tue 17:59: Instantiating kernel "SimpleKernel"
Tue 17:59: Compiling manager (CPU I/O Only)
Tue 17:59:
Tue 17:59: Compiling kernel "SimpleKernel"
Tue 17:59: Generating input files (VHDL, netlists, MegaWizard/CoreGen)
Tue 18:00: Running back-end  build (16 phases)
Tue 18:00: (1/16) - Prepare MaxFile Data (GenerateMaxFileDataFile)
Tue 18:00: (2/16) - Prepare for Placement (QuartusMap)
```



Hardware: logic programmable chip

- Less power consumption
- Faster
- 'Safer'

Meta compiler: MaxCompiler, Lava, lava (Haskell DSL), chisel, myHDL, TVM verilog output, ...

… And this is just to pitch an idea (1/2)

- Let’s discuss how we can solve this and get started!

Pitching Idea for Lane Detection Using DL

Pitching Idea

Some Starting Literature

- http://cs229.stanford.edu/proj2013/PazhayampallilKuan-DeepLearningLaneDetectionAutonomousVehicleLocalization.pdf
- http://www.cv-foundation.org/openaccess/content_cvpr_2016_workshops/w3/papers/Gurghian_DeepLanes_End-To-End_Lane_CVPR_2016_paper.pdf
- https://github.com/mvirgo/MLND-Capstone
- http://ocean.kisti.re.kr/downfile/volume/ieek1/OBDDBE/2016/v11n3/OBDDBE_2016_v11n3_163.pdf
- https://www.researchgate.net/profile/Vijay_John3/publication/281642917_Real-Time_Lane_Estimation_using_Deep_Features_and_Extra_Trees_Regression/links/55f256fc08aedecb6902120b/Real-Time-Lane-Estimation-using-Deep-Features-and-Extra-Trees-Regression.pdf

Current Limit of Lane detection

- Various lighting condition and road condition cannot be accounted for
- Perhaps with deep learning, we can train it with lots of different weather data

Two possible way

- Scene Parsing - > lane information (need find annotated scene parsed data)
- Traditional Lane Detection algorithm -> create data set ->train the model

Scene parsing : Segnet

- https://www.youtube.com/watch?v=CxanE_W46ts
- Get the road from scene parsing, connect them together

Traditional Lane Detection method

- https://medium.com/towards-data-science/lane-detection-with-deep-learning-part-1-9e096f3320b7

- But this is only when we have calibrated camera available

- Input is image, Output is polynomial coefficients, curves projected back to the screen

- Create Dataset, visually check all the videos. This will take up most of the time.

… And this is just to pitch an idea

- Let’s discuss how we can solve this and get started!