Realtime Detection using Deep Learning on Embedded Platforms

Summer Undergraduate Research Award

Arshdeep Singh
2016CS50625
Computer Science
CGPA: 8.49
Contact Number: 9582839784
cs5160625@iitd.ac.in

Mayank Singh Chauhan
2016CS50394
Computer Science
CGPA: 9.67
Contact Number: 7838298181
cs5160394@iitd.ac.in

Supervisor:-
Rijurekha Sen
Assistant Professor
Department of CSE
riju@cse.iitd.ac.in
IIT Delhi

Prof. S. Arun Kumar
Head of Department
Department of CSE
sak@cse.iitd.ernet.in

1 Motivation

In this project, we will investigate real time vehicle detection, counting and distance estimation from video streams using wearable and embedded cameras.

There are different application scenarios for this particular sub-system that fit well with other ongoing R&D activities in CSE department in IIT Delhi. A first use case is Assistech technology under Prof. M. Balakrishnan. Assistech aids visually impaired people with wearable cameras to ease mobility. For crossing roads both inside and outside the IIT campus, detecting vehicles and estimating their distance in real time is a very important task.

A second use case is vehicle mounted cameras to be used for Delhi-NCR pollution sensing with Ola cabs. This is a project undertaken by Prof. Rijurekha Sen with a plan to fit pollution sensors with on-board cameras in collaboration with Ola. The collected data will be used for fine grained correlation analysis of air pollution with road traffic congestion levels. Detecting and counting vehicles from the on-board camera feed in real time will greatly reduce the store costs for raw video footage on resource constrained sensing equipment or communication costs for the raw video footage to be transmitted to a remote server.

As a potential subsystem to integrate with the above two systems, we will investigate state of the art deep learning based computer vision models and customized deep learning hardware for embedded platforms, as part of our summer research.

There are different technical challenges to be tacked to solve this problem as listed below.

1. Object detection from wearable cameras is challenging due to difference in view points based on the height of the person wearing the camera, his or her physical orientation and physical movements. Assistech currently has a system to detect stationary objects like dogs sitting on the pavement with a wearable camera. But detecting an approaching vehicle would need much lower detection latency.

2. Vehicle detection using on-car sensors is a recent area of active research, thanks to self-driving car projects. Our target scenario for vehicle detection and counting on Indian roads is different in two aspects: due to cost constraints, our pollution sensing probe vehicles cannot be fitted with very expensive camera gear and additional ranging sensors. Therefore, detection needs to be done with cheap camera and inference hardware. Secondly, the type of vehicles and their nature of driving on Indian roads is very different from the self-driving car solutions developed for western traffic. Thus the object detection models have to be fine tuned with characteristic Indian vehicles like three wheeler auto-rickshaws.

3. There are multiple performance metrics to be optimized in addition to detection accuracy. For the wearable camera in Assistech, energy consumed while running real time detections has to be minimized to ensure longer battery life of the Assistech device. Detection latency is equally important, as that will ensure physical safety of the assisted person. Latency is also important for the vehicle mounted sensor, as

for a moving vehicle collecting sensor data, the view will constantly change needing the system to keep up with a good video frame processing speed. Energy is not that important for the vehicle mounted sensor, as it can take input power from the car's power outlet. Last, but not least important to consider is the (camera+inference hardware) platform cost, which needs to be minimized for both the target use cases.

Some other applications include improving latency in self-driving cars, detecting busy areas in street so that advertising bill boards can be priced appropriately, analyzing schemes like odd-even. The possibilities are endless.

Monitoring System	Information that can be collected	Advantages	Disadvantages
Computer Vision	1. Vehicle Detection 2. Count 3. Speed 4. Category 5. Path 6. Flow rate 7. Queue Length 8. Route Travel times 9. Lane changes	1. Lot of information can be collected 2. More than one lane can be monitored 3. Easy to install 4. Low maintenance 5. Autonomous 6. Easy to be networked	1. Not reliable in abnormal weather conditions 2. Field of view must be reasonably clear.
Magnetic Loop	1. Vehicle detection 2. Count	1. Reliable 2. Low maintenance 3. Autonomous	1. Dig up the road 2. Costly 3. Limited Information
Pressure tubes	1. Vehicle detection 2. Count	1. Reliable	2. Limited Information
Radar gun	1. Speed	1. Accurate speed measurement	2. Limited Information
Microwave sensors	1. Vehicle detection 2. Count	1. Reliable	1. Costly 2. Limited Information

Figure 1: Comparison of Computer Vision with other approaches

2 Objectives

Our main objective is to make and deploy a model capable of detecting and classifying *object of interests* in a video. The test time accuracy would depend on the presence of that object in the training dataset, this has an inherent problem that most of the datasets available online do not contain Indian vehicles for example auto-rickshaws. Also traffic behaviour is different in India compared to foreign countries.

The envisioned computer vision software will process an incoming video feed and do vehicle counting and classification (into types like buses and other public transport, cabs,

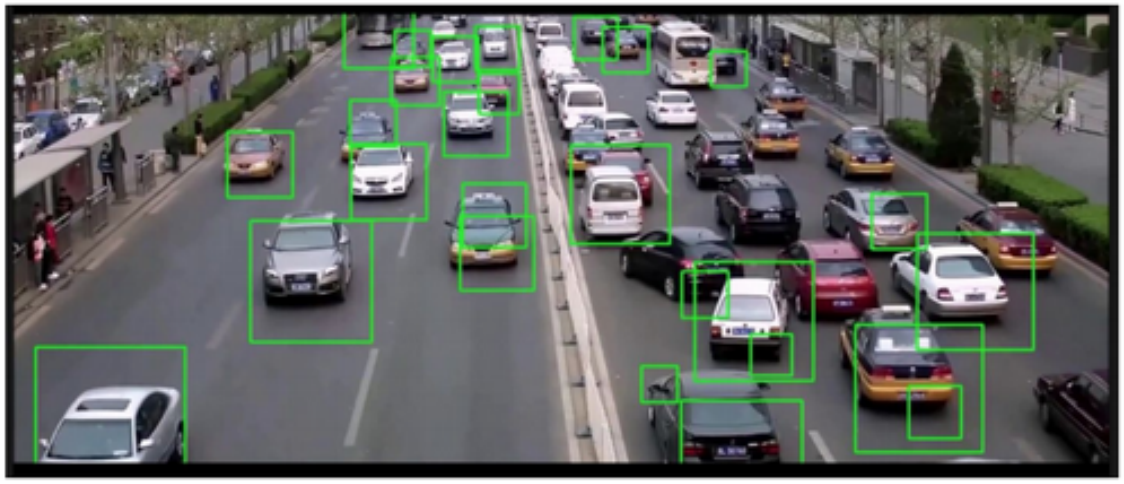


Figure 2: Detecting cars in an image

auto-rickshaws, personal cars, two-wheelers, heavy vehicles like trucks) in near real time. This information can be used in getting real-time information about traffic congestions, incidents and roadwork.

Processing high definition videos at powerful GPU servers might give the best performance in road traffic monitoring. But the poor broadband infrastructure in developing countries might prohibit real time streaming of HD videos from roads to servers. In-situ processing might mandate using mobile and embedded platforms, but their processor and battery constraints can conflict with heavy computation and low latency requirements. Our work would be to perform all the detection on embedded platforms like Intel Movidius sticks [13].

While accuracy of the inference task is an important metric to maximize, this might have trade-offs with other metrics on resource constrained embedded platforms. Is the latency of each inference too high to suit a real time mobile application while a user is interacting with it, or to suit a road traffic application to detect/prevent accidents? Is the trained deep-net model used in the inference too large to fit the embedded platform RAM? Does the inference task drain the mobile battery too fast? Our goal would be to see how different research communities are innovating to better handle these trade-offs [12].

To summarize it all, the objectives can be divided into following sub-tasks:

1. Organize all available sources of data and split them into training, validation and testing sets for detection and classification of *objects of interest*.

2. Implement several CNN architectures for object detection such as FRCNN [1], Yolo [2], SSD[3] for detecting objects in images.

3. Evaluate these architecture for accuracy, inference latency, memory usage and energy

consumption, based on a test fraction of the annotated dataset and perform modifications to use them on embedded platform.

4. Implement object detection from video in which objects' locations at each frame are required to be annotated with bounding boxes, based on still-image object detection and general object tracking bounding boxes and evaluate performance on embedded platform. [4]

5. Benchmark results and improve using techniques like Impression network for Video Object Detection [10][11].

3 Inference and Evaluation Platforms

While using GPU servers would give the best performance, but due to poor broadband infrastructure, realtime processing mandates using mobile and embedded platforms. Since their processor and battery constraints conflict with heavy computation and low latency requirements, we have a tradeoff between latency and accuracy. Our work would be to make an object detection system that is mobile, which is a challenge. We would have to select a detection architecture that achieves the right speed/memory/accuracy balance.

For evaluation purposes we would use Intel Movidius Neural Compute Stick [13] and NVIDIA Jetson TX2 Module[15]. Movidius, allows users to profile, tune, and deploy Convolutional Neural Network (CNN) on low-power applications requiring real-time inferencing. Movidius has specialized neural processing hardware with an interesting memory architecture[16] and is optimized for energy, hence can also be used as a wearable or on drones. Movidius Stick would be driven by Raspberry Pi[14], which can be mounted anywhere to inference task. NVIDIA Jetson TX2 Module is the fastest, most power-efficient embedded AI computing device available in today's market.

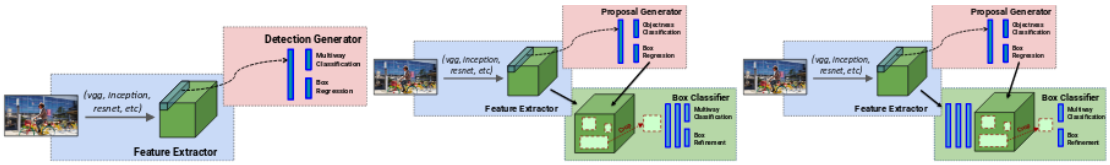
We would compare both the devices for accuracy/ latency/ energy or hardware weight which are important metrics for wearable computing and real time traffic density estimation.

4 Approach to the project

In this problem of deep learning based road monitoring, the computer vision algorithm has to detect *objects of interests* from a video frame. If the algorithm can output these detections for each video frame, vehicle and pedestrian counts will be statistics derived from those detections.

So the first thing that we'd do is detect objects from videos. We would begin by implementing object detection algorithm on images and evaluating our models on embedded platforms. Then we would extend this strategy to videos, in which objects of interest in each frame are to be annotated with bounding boxes based on still-image object detection and then evaluate performance on embedded platforms.

Overall the methodology is listed below:



(a) SSD. (b) Faster RCNN. (c) R-FCN.

Figure 3: High level diagrams of the detection meta-architectures

1. We would focus primarily on recent architectures: SSD (Single Shot Multibox Detector), Faster R-CNN, R-FCN (Region-based Fully Convolutional Networks) and YOLO. While they were presented with a particular feature extractor (e.g., VGG, Resnet, etc), we would experiment with various feature extractors

2. We would compare our results with standard benchmarks such as Imagenet and COCO with respect to accuracy.

3. In order to satisfy the constraints of the embedded platforms we could experiment with several architectural configurations, such as for Faster R-CNN and R-FCN, we can also choose the number of region proposals to be sent to the box classifier at test time. Typically, this number is 300 in both settings, but an easy way to save computation is to send fewer boxes potentially at the risk of reducing recall.

4. We would then be evaluating all the models based on accuracy/ latency/ energy or hardware metrics on Movidius Stick and NVIDIA Jetson TX2 Module.

5. An intuitive way to perform object detection in videos is to generate bounding box predictions for consecutive frames. We might need to drop frames in order to meet hardware constraints and improve latency.

6. Further Possibilities

   (a) We will look for techniques of data augmentation and transfer learning to compensate for the limited amount of training data.

   (b) We would look into using impression networks which might improve speed and performance simultaneously.

5 Datasets

1. PASCAL VOC: This dataset consists of realistic daily life images taken from Flickr. Intra-class variation is high and objects are viewed from multiple viewpoints. This dataset is composed of 20 object categories organized in four major groups.

2. MS-COCO: The COCO dataset is an excellent object detection dataset. It contains 91 common object categories with 82 of them having more than 5,000 labeled instances. Overall, the dataset has 2,500,000 labeled instances in 328,000 images.

3. Extensive video data on buses plying on various bus routes in Mumbai. This would mimic a real deployment setting [17]

6 Budget and duration

6.1 Budget

We would need to deploy our models for inference tasks on Intel's Movidius USB sticks which would be driven by Raspberry Pi. Apart from that we would also need to annotate unlabelled videos for which we'll use Amazon Mechanical Turk. All of this would cost around 25,000 rupees.

6.2 Duration

We would complete the object detection and classification by the end of the summer break i.e. the end of July. If time permits we may begin to research on ways to tackle the problems mentioned in motivation and would continue to work on them.

References

[1] *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks* arXiv:1506.01497

[2] *You Only Look Once: Unified, Real-Time Object Detection* arXiv:1506.02640

[3] *SSD: Single Shot MultiBox Detector* Proceedings of the European Conference on Computer Vision, 2016

[4] *Object Detection from Video Tubelets with Convolutional Neural Networks* CVPR, 2016

[5] Bengio Yoshua, LeCun Yann, Hinton Geoffrey *Deep Learning 2015*

[6] Angie K. Reyes, Juan C. Caicedo and Jorge E. Camargo *Fine-tuning Deep Convolutional Networks for Plant Recognition* LifeCLEF 2015

[7] Andrej Karpathy *CS231n Convolutional Neural Networks for Visual Recognition* http://cs231n.github.io/convolutional-networks/

[8] Santanu Chaudhury, Anupama Mallik, Hiranmay Ghosh *Multimedia Ontology: Representation and Applications*

[9] K. He, X. Zhang, S. Ren, and J. Sun, *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*, in IEEE International Conference on Computer Vision (ICCV), 2015

[10] Congrui Hetang, Hongwei Qin, Shaohui Liu, Junjie Yan, *Impression Network for Video Object Detection* arXiv:1712.05896

[11] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, Yichen Wei *Flow-Guided Feature Aggregation for Video Object Detection* arXiv:1703.10025v2

[12] *Speed/accuracy trade-offs for modern convolutional object detectors* arXiv:1611.10012v3

[13] Intel Movidius Neural Compute Stick https://developer.movidius.com/

[14] https://www.raspberrypi.org/

[15] https://developer.nvidia.com/embedded/buy/jetson-tx2

[16] http://eyeriss.mit.edu/

[17] Ravi Bhandari, Bhaskaran Raman, Venkat Padmanabhan *FullStop: Tracking Unsafe Stopping Behaviour of Buses*