Project Vita

Kishore | MIT

Contents:

Introduction:

Every year a lot of road accidents happen in India. Apart from fatal injuries, often people die due to many other issues like the ambulances arriving late or unable to arrive at the exact spot. This project aspires to evade this issue by helping medical personnel and first responders to make use of deep learning and apply this onto drones specialized for this specific purpose.

Setup:

- A drone with mounted cameras and an array of sensors including LIDAR, GPS etc.,
- A remote controller with video viewing capabilities
- NVIDIA TX1 development kit or suitable custom designed board
- DIGITS and TensorRT installed host

Working:

The victims of accidents need to be identified by the drone with aerial video input. When a road accident happens, the people on the spot call the emergency services. They often tell the operator only a rough location. This is a prime reason why the ambulances don't reach the spot on time. Also, in a rural scenario, there aren't enough landmarks to guide the vehicle to the spot. Now, these drones can be dispatched simultaneously with the ambulance. The drones shall make use of LIDAR and other sensors for navigator purposes. The operator can mark 1km radius based on the rough location through an interface. The drone shall also make use of the thermal cameras on board to scan the zone from a greater height in the air. A large number of heat signatures accumulated in a particular area may indicate the people crowding at the spot of the accident. These areas in the zone are given a high priority and closely scanned by the drone from a lower altitude. Once the drone finds these spots, it may make use of the Image-net trained data sets of pictures to identify the victims of the accident. Once it identifies them it gives the exact location of the spot to the ambulances using GPS. If there a large number of victims, it may signal the emergency services to send in more support vehicles.

This project can be further made more useful by linking it with an app. Almost everyone has a smartphone these days. Say, this project is being executed in a district level and all the people in the district are asked to install an app that captures some information like emergency contacts, blood group, previous medical history, medical conditions etc. Now, the drone on the spot, can query the mobiles of the victims for these data and send this to ambulances, database and even the hospitals so that they can receive some information about the incoming trauma beforehand. Of course, this will require the development of a whole ecosystem including the district emergency dispatch, ambulances, hospitals consisting of compatible devices and software.

The Drone:

The Drone should be sturdy, reliable and should be low-power consuming. It should be spacious enough to house all other components like the Tx1 GPU, a battery, various sensors etc. It should also be fast enough as time is critical in its usage.

Deep-learning Training:

The Drone's in-house GPU should be trained to identify victims in the spot. To do this, we train the neural network on-board the drones. The neural network maybe anything like AlexNet or GoogleNet. These neural networks can be trained using NVIDIA DIGITS. DIGITS is a contribution by NVIDIA and provides a platform to execute deep-learning training. ImageNET is an open source project that contains millions of pictures of numerous categories.

ImageNet – identifies/classifies the image

detectNet – object localization

segNet - accepts a 2D image as input, and outputs a second image with the per-pixel classification mask overlay. Each pixel of the mask corresponds to the class of object that was classified.

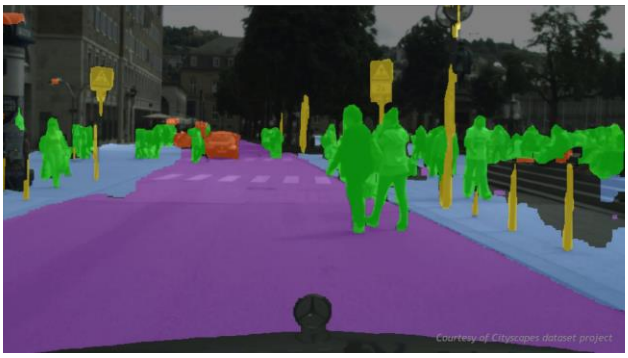
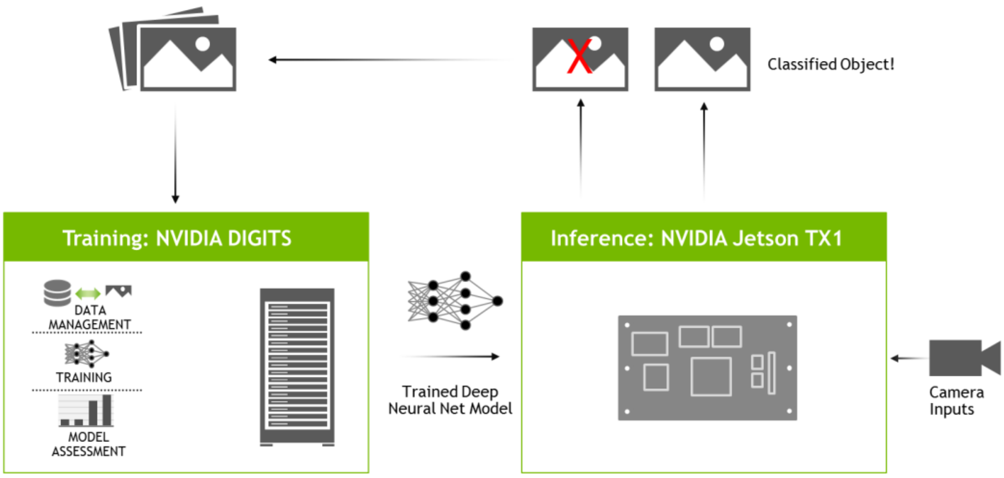


Image Recognition
Classification

Object Detection
Localization

Segmentation
Free Space

Examples are provided for streaming from live camera feed and processing images from disk.



Throughout training, the network's inference performance is tested and refined using trial dataset. Like the training dataset, the trial dataset is labelled with ground-truth so the network's accuracy can be evaluated, but was not included in the training dataset. The network continues to train iteratively until it reaches a certain level of accuracy set by the user.

Due to the size of the datasets and deep inference networks, training is typically very resource-intensive and can take weeks or months on traditional computer architectures. However, GPU’s are able to parallel-process and thus reduce the time. But the task for the GPU should be bulk-enough to hide the latency issues caused by them. So, using GPUs vastly accelerates the process down to days or hours.

Deep-learning Inference:

With TensorRT from NVIDIA for deep-learning inference, we deploy a deep learning application onto a GPU, increasing throughput and reducing latency during inference. Deep-learning inference is using the datasets provided during training and applying that information during real-time usage. The DNN (Deep Neural Network) for this project is best suited to have a greater number of hidden layers. A greater number of hidden layers will require a larger amount of time to train the neural network and but will result in higher precision and accuracy which is essential in out usage.

The Builder in TensorRT builds an engine takes a network with a pre-trained model and generates an engine that is optimized for the target platform. The Engine is the one which takes the input and performs inferences and displays outputs.

Why GPU (Nvidia TX1) rather than a CPU?

This project uses an NVIDIA TX1 chip which is a GPU. Why not a traditional CPU? CPUs are faster than GPUs when handling a relative low intensive task. This is due to the larger cache memory in CPUs due to which data transfer is faster. In a GPU, throughput matters. So, the cache memory is reduced and the transistor count is increased for this very specific purpose. Hence, with an intensive tasks, such as deep-learning inference in our case, a GPU is suitable. The latency caused by lower cache memory is masked by the intensity of the task and parallel algorithms.

Resources:

1. https://github.com/dusty-nv/jetson-inference/blob/master/docs/deep-learning.md
2. https://developer.nvidia.com/embedded/twodaystoademo
3. https://github.com/dusty-nv/jetson-inference#system-setup
4. https://github.com/dusty-nv/jetson-inference#image-segmentation-with-segnet
5. https://github.com/dusty-nv/jetson-inference#classifying-images-with-imagenet
6. https://github.com/dusty-nv/jetson-inference#locating-object-coordinates-using-detectnet
7. https://developer.nvidia.com/digits
8. https://devblogs.nvidia.com/speed-up-inference-tensorrt/

9. NVIDIA Webinar (https://www.youtube.com/watch?v=nRSxp5ZKwhQ&index=2&list=PL3MC2p4fFAD1iIWgZ13mSj0c2xoAmH4A6&t=2148s)