Unified Transport Engine: Learning to Predict Traffic Congestions

Ramesh Sarukkai, Shaohui Sun
Mapping & Perception
Lyft Inc.
Palo Alto, CA
rsarukkai@lyft.com, psun@lyft.com

Abstract

We present the framework for an unified transport engine that allows for streamlining wide variety of data sources and designed to support different elements required for seamless transport ranging from speed profiles, congestion, estimated time, features required for routing, and so on. In this paper, we specifically dig into the problem of real time traffic congestion estimation using imagery collected from standard mobile phones. We discuss how we can quickly build highly accurate detectors using transfer learning. Examples and edge cases that worked well as well are challenges for our system are presented.

1 Introduction

Transport in the physical world is an incredibly hard problem, even for humans. There is a lot of complexity ranging from perception (lane, signage), localization (GPS, SLAM), marketplace dynamics (supply/demand), accurate maps/routing data, passenger/driver personalization, and much more. This problem is compounded by the fact that there is noise and lossy compression happening at various layers. Its incredible that billions of people are able to drive relatively safely on a daily basis!

1.1 Classic Approaches

Classic approaches generally have been to collect data directly from vehicles - either through embedded sensors (OEMs) or through mobile devices. In either case, the data is pre-processed locally and then aggregated in the backend. A series of machine learning and filtering algorithms are applied server side in order to determine various transport related data such as traffic congestion, speed, vehicle and passenger localization and so on. This data is then used to execute on functionality required - such as dispatching a driver for a passenger request, auction in the marketplace to find the closest vehicle, expected time to reach destination or pickup and so on. Currently, classic non-Autonomous scenarios for mapping and ride sharing don't use imagery for such functionality.

1.2 Autonomous Vehicles

With the advent of autonomous vehicles, it's critical to be able to learn and scale solutions to these problems automatically - especially using imagery. With the recent spurt of storage and compute resources, we are able to do more and more with data - pushing the envelope to the edge, but many fundamental problems in putting this all together remains. There are two variations thematically for driving autonomous vehicles - one end of the spectrum is end-to-end driving where the model suggests required action (e.g. steering wheel angle) from imagery/sensory inputs. Another end of the spectrum is a more semantic approach where we perceive objects in the world, have a refined understanding of the environment and then have a planner to parse and navigate the vehicle. In either scenario, it's useful to have context of the world (as it relates to transport) such as road blockages, traffic speeds and accidents as they are vital to ensure timely and optimized routing plans.

1.3 Related Work

There have been a number of papers with resurgence of imagery based learning for autonomous vehicles[9]. However, the efforts are more focussed on imagery to action (steering wheel and so on)t using deep learning approaches. No research emphasis has been placed on an integrated solution that puts together different elements of transport and tying them together.

On problems related specifically to traffic density, Shanhang et al [4] report on an approach to understanding traffic density from web data. Their effort is focussed on web cam data instead of real time traffic congestion using real time data from vehicles (ride sharing or autonomous). On the deep learning approaches side, solutions have typically tried to solve object counting as applied to crowd density estimations [6,7].

In contrast, our work here focuses on vehicular congestion based on real traffic imagery contrasting to the related work which was targeting people counting problems. Additionally, we present a transfer learned approach for this problem to accelerate training. Furthermore, the approach of breaking down component wise different transport problems is a scalable and cohesive approach to train and reuse across multiple problems related to transport.

2 Unified Transport Engine (UTE)

As described earlier, there are a number of ad hoc solutions in place that power existing mapping and ride sharing marketplaces. There has been a lot of success in deep learning where a number of different functional units are combined together to build a more comprehensive system. The goal of our work is to investigate an unified approach to transport using deep learning. Such an approach allows for knowledge transferability across different components, as well as sharing signals across the whole system. By having a unified framework, we ensure consistency, common data sets for training, and the opportunity to do joint optimization across systems. In particular, we show different components that are critical to transport such as routing, ETA, dispatch and so on. The ability to jointly train and cross functional boundaries is an important step towards unification of a transport engine. Furthermore, this allows for more flexibility in expanding to new sensory data - such as transition to leveraging imagery across different systems.

The overall architecture of the Unified Transport Engine is shown in Figure 1. At a high level, the key components are summarized below:

2.1 Input Streams

Fundamentally, the input stream consists of a set of different sources of sensory and historical model data:

● **Mobile Data**: This refers to set of sensory information collected from the device - e.g. GPS data collected from mobile phones. This is often processed/filtered through a variety of algorithms before being aggregated into the feature set.
● **Camera Data**: Imagery is an integral part of Mapping and Autonomous stack. Data can be captured via special cameras mounted on the car or through mobile phones as in the case of crowdsourcing. Number of variants here depending on the objective/use of the data - sequences of image frames are required for extracting motion/structure primitives, single images for perception and so on.
● **Lidar/Radar Data**: Autonomous vehicles today typically collect additional non-sensory data using Lidar, Radar, or Sonar Systems. These are generally more robust to estimate depth maps of the world but often come with significantly additional costs.
● **Priors/Historical**:Another important ingredient is data collected from prior experience - billions of rides, usage patterns and hotspots - this data is run through separate systems and machine learned models

and often condensed into core, interesting feature sets.

- **Mapping Data**: Another essential ingredient is the base and semantic map - which serves as the foundation for other transport algorithms that operate on top off and whose accuracy dictates the accuracy of systems downstream that utilize the data (e,g, for routing).

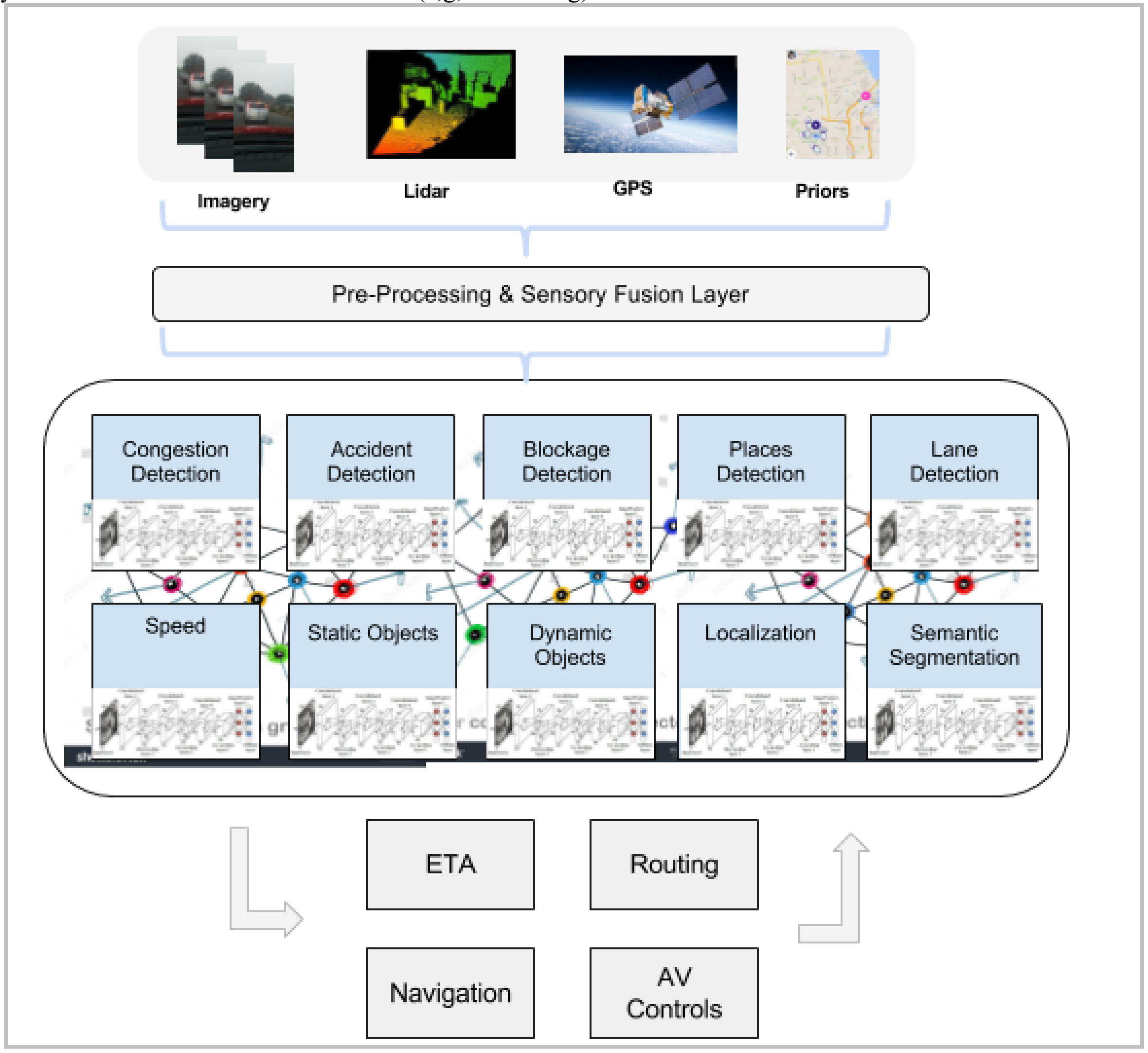

Figure 1: Unified Transport Engine

2.2 Functional Sub-Systems

First, we cover the functional sub-systems within our Unified Transport Engine:

- **Congestion detection**: Detection of congestion - this could be based on density of vehicles, combination of differential distance estimates, and so on. We cover this in detail in the later part of this paper..
- **Accident detection**: These are training for patterns where the situational data doesn't match normal traffic scenarios - Cars with open doors in middle of highway, or passenger/drivers standing out in a dense set of cars and so on. Data gathering is harder for this scenario, but with proper human selection and labeling of data sets, we get sufficient coverage.
- **Blockage detection**: This refers to incident detection related to situations like temporary road blockage and construction activity.
- **Speed Estimation**: Estimating speeds directly from multi-sensory data - this can be map-matched to the lane level as well. Some of the challenges here is that we need to accommodate different cases such as

HOV lanes, traffic density signals derived from congestion and so on.

- **Lante Detection**: Typically needed for autonomous vehicles - getting high level of accuracy of lanes and orientation is critical.
- **Perception of static and dynamic objects**: This is a fairly large sub-component - with a combination of recognizing static objects (cars, people)

and so on. This list is not meant to be comprehensive, rather meant to capture examples of functional modules that we embed within the scope of the universal transport engine. The aforementioned components then feed in and integrate with "classic transport" systems like dispatch and scheduling ultimately to provide the transport service for specific rides or customers.

3 Experiments

In this section, we review our experimental results. We have focussed the work in this paper on primarily imagery and a specific functional module described earlier - namely traffic congestion detection. Congestion detection is an important problem that feeds into traffic and routing algorithms as signals. This also gives us a good specific use case where we can understand and validate solutions as well as identify potential problems.

3.1 Imagery Data Collection

In order to collect imagery for this specific experiment, we chose the simpler path of using imagery from mobile phones**.** We mounted an ordinary consumer phone behind the windshield to capture images while driving. The test area is a typical urban area around the city of Palo Alto in California, United States. In this exercise, we collected a dataset of over 8000 images, and used a subset of labelled data for training, cross-validation and testing. Our subset of data was chosen to ensure sufficient coverage of scenarios of both clear and congested traffic in diverse conditions (small streets, parking lots, highways/routes). The preliminary goal is to conduct the first attempt that is part of our grand vision in which every driver should be able to contribute to our universal transport engine only using their phones or similar portable consumer devices. The techniques are expandable to data sets collected in other means such as cameras mounted on autonomous vehicles.

3.1 Classifier trained with transfer learning

For the congestion module, we decided to experiment with a end-to-end solution using the state-of-art of neural network models. PyTorch [3] is the core machine learning library / platform used here [5. The whole dataset is divided into the training set, the validation set and the test set. We conducted transfer learning using a variety of ResNet [8] pretrained models that are already provided by PyTorch. We modified the last fully connected layer to fit the specific purpose here. While training, the data are randomly transformed for the purpose of data augmentation. We only optimize the parameters of the last layer since we do not have a big enough dataset. Doing so is the common practice to avoid over-fitting. We run all the models for 25 epochs. The learning curve from ResNet101 is shown in Fig. 2.

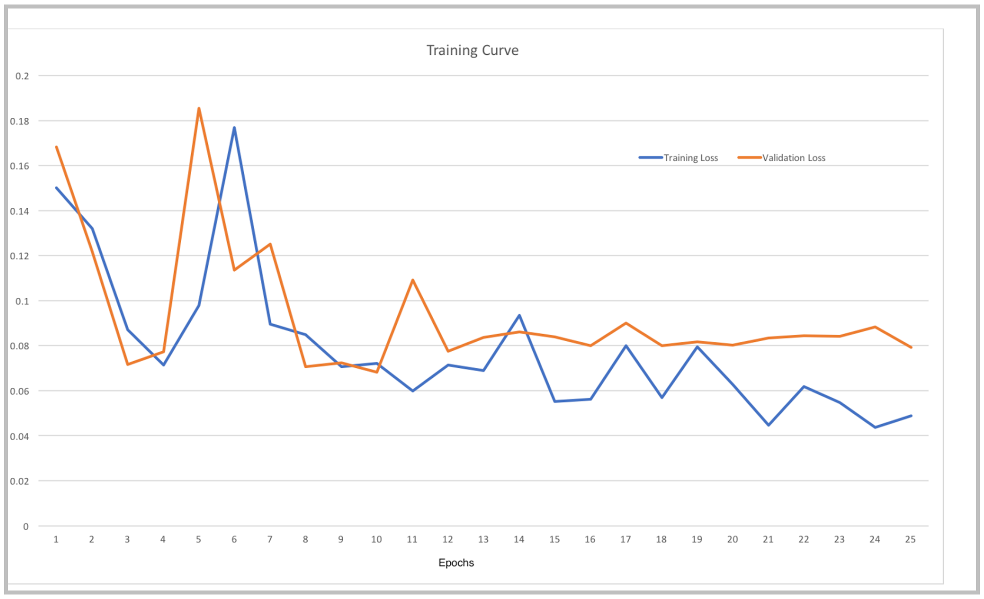

Figure 2: Training Curve (Resnet101)

3.1 Variance of imagery collection poses

At the very beginning, it was worrisome to us that in most of the images, a significant portion of one image is actually the image of the hood of the vehicle. We first cropped images in a calculated way trying to reserve the actual traffic scene only. However, this turns out to be unnecessary. As a matter of fact, in reality, we can not control each individual driver on how they mount their phones since we have no plan to convince drivers to use the same mounting setup. It is extremely difficult to scale in various ways.

The accuracy of our model can be seen in Table. 1. As shown, it can achieve the high accuracy as much as about 97%. By far, we have not seen the advantage of using a deeper network against a shallower one. Our belief is that the expected advantage should surface once the volume of training data grows exponentially.

Table 1: Performance using different models

Iodel	**ccuracy**
esnet18	.79%
esnet34	.97%
esnet50	.74%
esnet101	.88%
esnet152	.93%

4 Discussion

4.1 Model Interpretation

Next we look at some of the anecdotal experiences with the results. Figure 3 shows examples of imagery and the prediction by one of our classifiers. Specifically, incorrect predictions are bounded in red boxes. There are a few patterns we noticed:

● The model trained to look for a number of features related to the back of cars/vehicles - this seems to be the right thing to do. However, for the non-congestion case, a number of training samples had low trafficked streets - hence exposing a lot of road in the imagery. In some of the examples shown, we see that even though congestion can be seen ahead (e.g. 40 feet ahead), the gap in the open road seems to trigger mis-recognition as the non-congested scenario.

● It is quite fascinating to observe that in some cases, the traffic is not congested towards the driving direction, but it is on the opposite direction. We consider it as a clear traffic situation relative to the driver. The system is able to identify this kind of challenging case.

● Other patterns that were a bit tricky were carpool lanes - where the traffic is flowing immediately ahead but overall imagery suggests congested traffic - we are looking to improve this by combining with other sensory (localization) and mapping (lane) data. The results overall look promising and future work is focussed on integrating this back into the actual marketplace for estimating pickup, dropoff and traffic speeds.

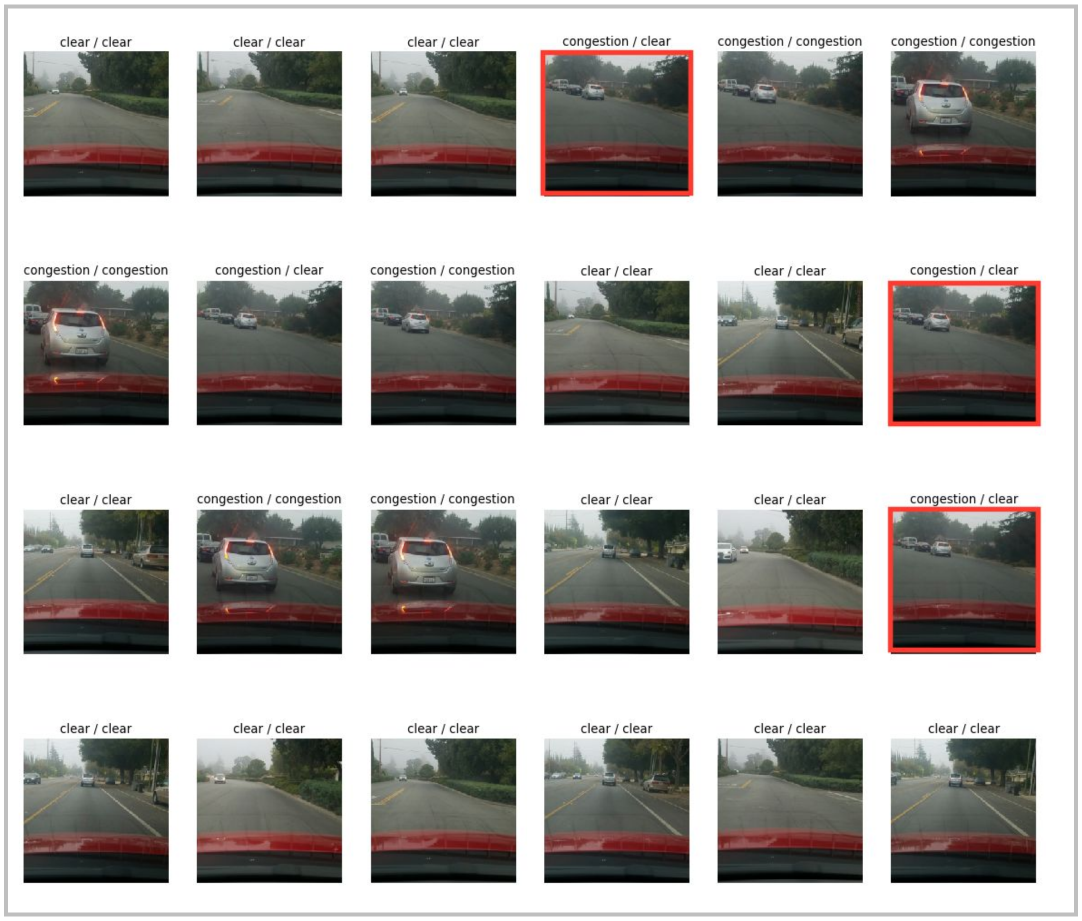

Figure 3: Results on a small number of samples (each sub-caption is formated as {ground truth}/{predicted}), wrong predictions are highlighted in red rectangles

4.2 Expanding use cases for transport

It’s important to preface the discussion of the results by emphasizing that traffic congestion is a material use case today in navigation and transport systems - one of the more reliable signals of congestion is from social crowd-sourcing applications such as Waze which have anecdotally proven to be very beneficial to improving safety, efficiency and comfort of rides. We also have integration with imagery with maps (see Fig 4) that allows review and verification of incidents/events using this derived data.

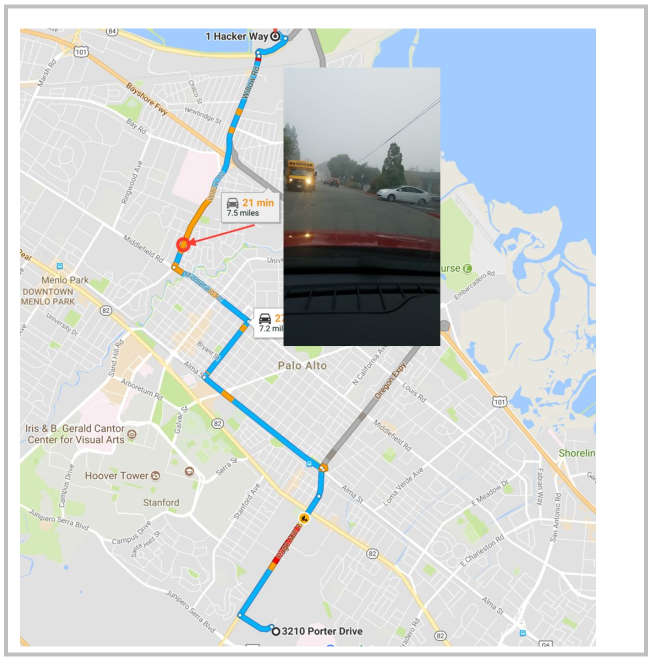

Figure 4: Local congestion recognition from crowdsourced imagery can help provide more accurate and timely updates to the global map (note: this is only a demonstration, and the photo shown and its relative geolocation are not necessarily matched) (courtesy to Google Maps)

References

[1] Tara N. Sainath and Carolina Parada. 2015. Convolutional Neural Networks for Small-Footprint Keyword Spoing. In Proceedings of the 16th Annual Conference of the International Speech Communication Association (Interspeech 2015). Dresden, Germany, 1478–1482.

[2] Shanghang Zhang et al, Understanding Traffic Density from Large Scale Web, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17), Hawaii.

[3] Raphael Tang and Jimmy Lin, Honk: A PyTorch Reimplementation of Convolutional Neural Networks for Keyword Spoing

[3] PyTorch, http://pytorch.org/

[4] Martin Wirz et al, 2013, Probing crowd density through smartphones in city-scale mass gatherings, EPJ Data Science 2:5

[5] Deng, J., Dong, W., Socher, R., Li, L., Li, K., & Fei-fei, L. (2009). ImageNet : A Large-Scale Hierarchical Image Database. In IEEE Conference on Computer Vision and Pattern Recognition.

[6] C. Zhang, H. Li, X. Wang, and X. Yang. Cross-scene crowd counting via deep convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 833–841, 2015. 2, 8

[7] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma. Singleimage crowd counting via multi-column convolutional neural network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 589–597, 2016. 2 [37] Z. Zhao, H. L

[8] Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2016). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv preprint arXiv:1602.07261.

[9] Mariusz B et al, 2016, End to End Learning for Self-Driving Cars, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '16).