CUSP-GX 5006.001
Professor Daniel Neill
30 May 2018
Julian Ferreiro, julian.ferreiro@nyu.edu
Lingyi Zhang, lingyi.zhang@nyu.edu
Zhiao Zhou, zhiao.zhou@nyu.edu

Traffic Count Detection

Abstract

Traffic count is the fundamental element of transportation-related studies. Although traffic counting has been widely studied, more effort is on annually traffic estimation based on collected daily traffic counts, rather than improving accuracy and efficiency of the raw data collection. Moreover, most of people are still collecting traffic density data manually instead of using machines. We successfully implemented two CNN-based models (classification and object detection) for detecting vehicle numbers in images: Inception Resnet V2 (Model 1) and Faster RCNN with Inception Resnet V2 (Model 2). In model 1, Inception Resnet V2 was used to classify the vehicle counts ranging from 0 to 41. In model 2, we first conducted detection of cars using Faster RCNN with Inception Resnet V2 and then count the frequency of detections in each image. In this way, traffic counts can be conducted automatically and in real-time using pre-installed traffic cameras, which could largely reduce expenses on labor as well as extra equipment installation and maintenance.

Introduction

Maintaining mobility is a priority for cities to be viable. A transport system within a city needs to efficiently connect jobs with households to ensure its growth and productivity (Bertaud, 2014). With the growing number of people moving to cities, the study and monitoring of transportation are now more relevant than ever.

In the past traffic flow has been studied as an important factor that influences commuting time, the occurrence and fatality of road accidents (Vickerman, 2000), and needs to be considered to choose where to locate pedestrian crossings, traffic signs, or how to redirect traffic in case of an

accident or construction. In effect, the goal of understanding traffic flow is to make cars traverse the highest distance in the shortest amount of time possible.

Traffic flow can be characterized by the speed, density, and flow of traffic (Daganzo, 2008). These macroscopic variables are commonly computed using averages of the whole traffic stream (Highway Capacity Manual, 2000). However, traffic density (the number of cars in a specific road segment) is difficult to estimate in this way and is usually inferred from the flow rate and the speed (Highway Capacity Manual, 2000). For this reason, providing a methodology to acquire the counts of cars in a specific road segment could add more granular information to describe and model traffic patterns.

Our goal is to train a Convolutional Neural Network (CNN) to count the number of cars in a road segment. We find this project interesting for the following reasons: First, complement the knowledge learned in class with a novel technique in computer vision. Second, create data. Although we live in an era where information is readily accessible, many times the existing data is not enough to approach a problem or some cities may not have developed the data collection process that makes the data available. In these cases, new data must be generated to aid in the resolution of the problem. CNN can provide a framework to solve this lack of data since it can be deployed using existing technology that is more readily available in most cities, such as traffic cameras. The final reason is more epistemological in nature. What we are currently pursuing is not novel and has already been done with success. We thus aim for reproducibility. The researchers in the paper we are basing our work created and labeled their own dataset. Lacking the resources to do this, we will use publicly available datasets to train our own CNN and determine if we can achieve similar results using a lower quality dataset.

Current traffic count methods in practice mostly rely on human work. Take the New York State Traffic Monitoring Standards as an example, manual counts require a minimum of 15 minutes intervals as a valid count (NYSDoT, 2018), which is labor and time inefficient. They do also start using non-intrusive count methods like radar or infrared types of equipments. However, this will require extra expense for equipment installation and maintenance.

The use of Machine Learning to automatically detect traffic patterns has multiple benefits, it makes the data collection process less time consuming for humans, allowing to get the data without the need of using human hours and it also allows the continuous monitoring of the variables of interest, allowing a higher granularity of the data. Meanwhile, it can use readily available technology in cities, such as traffic cameras, to get the information, making the process of deploying the technology more attractive to city agencies.

The approach of using computer vision is not novel, and has been used in the past (Sohn, 2016). There are multiple methodologies to perform this task and we will try to determine the uses and limitations of some of them in the current work.

Data and Methods

Two different datasets have been employed in our research: the Cars Overhead With Context (COWC) and the Annotated Driving Dataset. We first used COWC as the training set, and experimented with a four hidden layer convolutional neural network (regression) and a Capsule Networks (regression), which both ended with unsatisfactory results. Eventually, we successfully implemented Inception Resnet v2 (Model 1) for classification. Although Model 1 works perfectly with COWC, with accuracy higher than the model Mundhenk, Konjevod, Sakla, and Boakye (2016) had built, we found out that as the training data comes from an overhead views, the model we have trained might not work well with camera views. As a result, we found the Annotated Driving Dataset and successfully trained a new model Faster RCNN with Inception Resnet V2 (Model 2).

Dataset

COWC is an annotated dataset of overhead car photos (Figure 1a) created by a group from the Computation Engineering Division at Lawrence Livermore National Laboratory. COWC includes images from cities (Toronto Canada, Selwyn New Zealand, Potsdam and Vaihingen Germany, Columbus and Utah United States). All images are at 15 cm per pixel resolution at ground (Mundhenk, Konjevod, Sakla, & Boakye, 2016). In total, there are 32,716 unique annotated cars. 58,247 unique negative examples. For the data preprocessing, we cropped the grey frames of all images. We dropped the data of Utah, which contains almost 60% of the original dataset because out the hard disk in the virtual environment for our analysis. As data augmentation has been done in the original dataset (Figure 1b), no further data preprocess was conducted. The labels in this dataset only correspond to the number of cars present in the image and go from 0 to 41.

Figure 1. Training Data for Model 1. (a) They were labeled as 26, 10, 0, 6, 1, 4 respectively. (b) Data augmentation with the original datasets.

The choice of using a second dataset was done for two reasons: 1) the lack of additional labels in the COWC dataset made the task of counting more difficult and 2) the viewing angle in the

COWC dataset is not the most frequent one seen in traffic cameras and this would affect the applicability of the resulting CNN.

The Annotated Driving Dataset (Gonzalez, Higgins & Cameron, 2017) included 24,423 frames with labels in total collected from a Point Grey research cameras. These images are taken from a more horizontal viewpoint (Figure 2a) compared with the dataset we mentioned above. All cars, trucks, and pedestrians in each frame are annotated with a bounding box (Figure 2b). This dataset includes bounding boxes and labels that identify cars, trucks, bikers, traffic lights and pedestrians.


(a)

(b)

Figure 2. Training Data for Model 2. In this datasets, objects are annotated with bounding boxes.

Methods

We tried 4 different models: CNN with four hidden layers, Capsule Networks, Inception Resnet v2, and Faster RCNN with Inception Resnet V2. The latter 3 of them are adopted from pre-built architectures. All of them are trained using Tensorflow with a Nvidia P5000 GPU. Regression and detection were performed over the COWC dataset, while detection was done over the Annotated Driving Dataset.

1. Regression

We started with regression using the very basic CNN first, which includes 4 hidden layers. The result is not promising with an accuracy of 23.2%. We then switched to Capsule Networks. This method is good at preserving the hierarchical pose relationships between parts of an object (Sabour, Frosst, & Hinton, 2017), which can keep image details and handle ambiguity so we adopted its architecture (Figure 3) on the COWC dataset. The results were not good enough for the regression model since no matter how the hyper-parameters were tuned, the loss function was not converging.

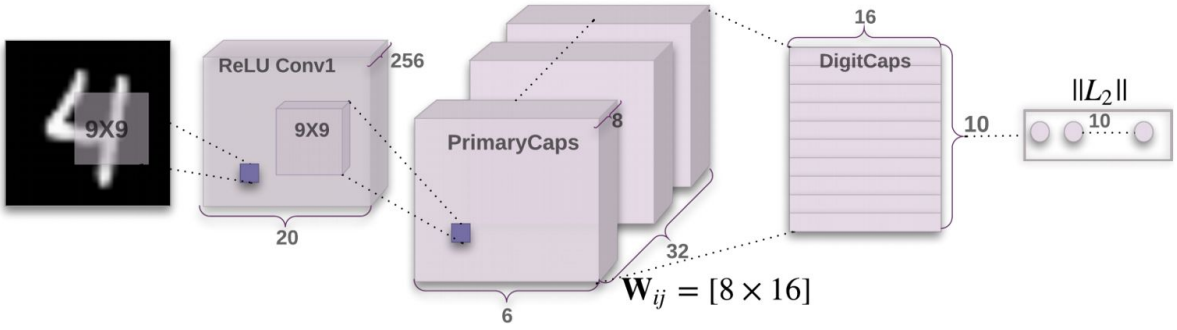

Figure 3. Capsule Networks Architecture.

2. Classification

Although there are instances of using CNN for regression (Walach & Wolf, 2016), this approach did not prove fruitful for our goal, a small neural network cannot successfully extract the features of tiny objects in images of the COWC dataset. CNN seems more widely used for its ability to do classification, so we adopted the Inception Resnet V2 for classification which is now one of the most accurate and state-of-art classification algorithms as it gets the best performance in ILSVRC image test, developed by Google in 2016 (Alemi, 2016). Compared to the Capsule Networks, Inception goes even deeper and complex (Figure 4) for achieving better performance on image recognition. Meanwhile, the introduction of Resnet further guaranteed the performance of very deep networks (Szegedy, Ioffe, Vanhoucke, & Alemi, 2017).

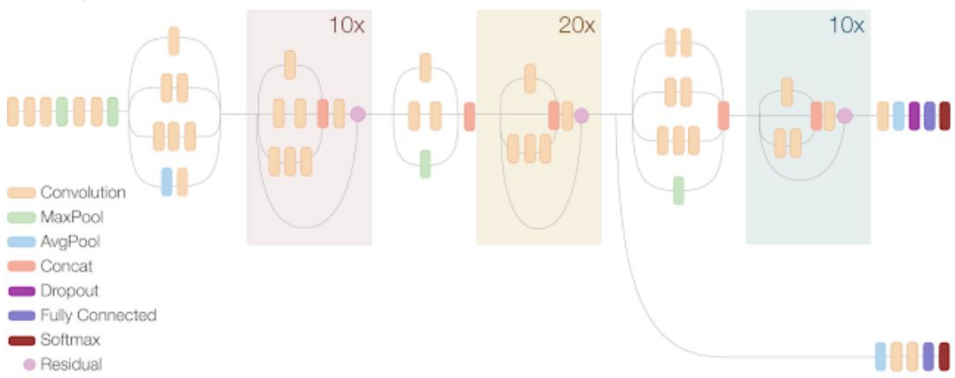

Figure 4. Schematic Diagram of Inception-ResNet-v2

We trained our model on 20k iterations with an initial learning rate of 0.1 and decay rate of 0.5. However, unlike what Google does by default, we used a cyclical learning rate to make the loss function keep decreasing on a steady basis. What's more, we tried to finetune all of the layers in the network instead of using ones Google provides because of the difference of our dataset with

theirs. It turns out that it has a high recall and precision rate as shown in Table 1 which means that it classifies correctly on those classes that it needs to classify. However, the accuracy rate is not high due to an “empty car” class we added before training the model in case there were test sets that had a numbers of cars which were not in our classes. An accuracy rate of 84% means that the classifier classifies quite a few samples in the empty class which leads to our concern that is whether a classifier can be powerful enough to detect features in the image especially these ones containing only “half” cars.

Table 1. Model 1 Performance

Model	Accuracy	Recall	Precision
Inception Resnet V2	84.12%	99.77%	98.37%

3. Object Detection

That concern results in our next step which is to train an object detection model because we think that with specific bounding boxes, the model can detect the vehicles more accurately and more easily. This time we did not just select the most accurate learning algorithm because we have to sacrifice the accuracy to relatively lower the cost and time. Consequently, we chose Faster RCNN with Inception ResNet V2 Inception Resnet V2 showed a promising result for car classification. For car detection, we chose the Faster RCNN with Inception Resnet V2 (Ren, He, Girshick, & Sun, 2015). And more importantly it’s large and powerful enough for detection and it only takes 0.7s to run on one iteration. We set all of the hyper-parameters by default and tuned only the last softmax layer or it can take nearly forever to make the lost function converge. After 22k iterations, we got a mean average precision of 0.8 on our test set. An example is shown below.

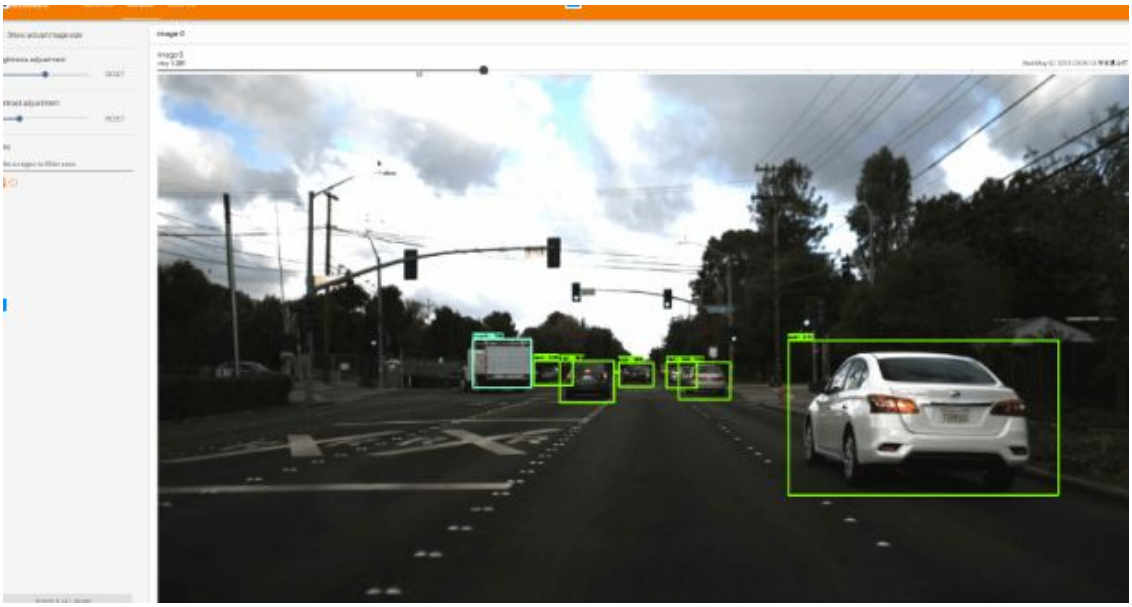

Figure 5. Example of Faster RCNN with Inception ResNet V2

Last but not least, we tested our two models on a few real-time traffic cam streams from Manhattan, New York City. It turns out that using classification on them can be mighty slow while the object detection works pretty well.

Conclusions

Convolutional Neural Networks are powerful and useful methods that can be adapted to existing technology and datasets, generating insight in an automatic way and also freeing humans to do other tasks.

However, these methods have limitations. For a start, they require a vast amount of labeled data. Since our project involved common elements of the urban landscape (i.e. cars), it was not difficult to find available datasets. Nevertheless, the available dataset included overhead images of cars that to our knowledge are not that common in traffic cameras (see Appendix). In much of the literature we reviewed, researchers that achieved good performance also generated their own datasets. This shows the importance of data collection and posterior labeling that these methods require. If researchers or public agencies are interested in applying these type of methods, these constraints should be taken into account. Researching available datasets in the the topic of interest is a critical part of establishing a pipeline for this type of project. Also, although the algorithms are open, the specific type of data they require could involve a lot of work (e.g. labelling by human operators) that could be time and resource consuming in a real world setting. For this reason collaborations with institutions in other geographies could benefit equally all of the parties and reduce the costs of this methodologies.

Another conclusion of this work was, even if we had a relevant labeled dataset, a lot of fine-tuning and knowledge about how the algorithm works are required. One relevant example we encountered was while using the *Inception Resnet v2 model* we first came across low levels of accuracy and a high loss function. This was due to that by default, the algorithm was designed to crop the images to augment the original dataset (artificially expanding the dataset) and reduce possible overfitting. When combined with the COWC dataset, that already contained cropped images, this made the training set very inefficient for the algorithm to detect cars and was resolved by removing this crop functionality embedded in the algorithm.

Another technical limitation was that the type of label limits the available methods that can be applied to. Having only the car numbers of an image do not enable to do object detection, this needs a labeled dataset with bounding boxes labeling the objects of interest. Consequently, this will also limit the performance of the model.

It was also interesting for us to encounter methodologies in the literature that were not trivial to replicate. Although the paper detailed the algorithms used, and the performance they got, the specific steps and algorithm were difficult to extract from the paper and the available online

information. Although our purpose was not to do exactly the same, a lot of researching was required to achieve similar results.

Finally, compared to the other machine learning methods we were used to, CNN requires much less data wrangling and manipulation but much more parameter tuning. This is required both a much deeper understanding of how the algorithm works and how the parameters affect the results.

As for the implications of our work, the experience was useful to have a first approach to CNN models. Although they are time-consuming and require much more fine tuning than we originally expected, the results we obtained made the experience very rewarding.

Team Contribution

Julian: Literature review, conclusion, dataset research.
Lingyi: Methodology, scope definition, dataset research.
Zhiao: Methodology, coding and fine optimization of the models.

The team wishes to disclose that the technical work could not have been done without the research in methods and the coding that was performed by Zhiao and that without his work we couldn't have achieved neither the levels of accuracy nor the depth of understanding in CNN methods.

Reference

A Beginner's Guide To Understanding Convolutional Neural Networks - Retrieved from: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/

Alemi, A. (2016). Improving Inception and Image Classification in TensorFlow. Retrieved from: https://research.googleblog.com/2016/08/improving-inception-and-image.html

Bertaud, A (2014). *Cities As Labor Markets*. Retrieved from: https://marroninstitute.nyu.edu/uploads/content/Cities_as_Labor_Markets.pdf

Cars Overhead With Context (2016). Computation Engineering Division at Lawrence Livermore National Laboratory. Retrieved from: https://gdo152.llnl.gov/cowc/

Chung J. and Sohn K., "Image-Based Learning to Measure Traffic Density Using a Deep Convolutional Neural Network," in *IEEE Transactions on Intelligent Transportation Systems*.

Dickerson, A., Peirson, J., & Vickerman, R. (2000). Road Accidents and Traffic Flows: An Econometric Investigation. *Economica, 67*(265), new series, 101-121.

Geroliminis, Daganzo (2008). *Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings*. Transportation Research Part B: Methodological, Volume 42, Issue 9

Gonzalez, E., Higgins, M., & Cameron, O. (2017). Annotated Driving Datasets – Many hours of labelled driving data. Retrieved from: https://github.com/udacity/self-driving-car/tree/master/annotations

Highway Capacity Manual 2000. Transportation Research Board. National Research Council of the United States of America.

Mundhenk, T. N., Konjevod, G., Sakla, W. A., & Boakye, K. (2016). A large contextual dataset for classification, detection and counting of cars with deep learning. Paper presented at the European Conference on Computer Vision.

New York State Traffic Monitoring Standards for Short Count Data Collection (2018). New York State Department of Traffic. Retrieved from: https://www.dot.ny.gov/divisions/engineering/technical-services/hds-respository/Tab/NYSDOT_Traffic_Monitoring_Standards_for_Short_Count_Data_Collection_EB_18-005.pdf

Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster r-cnn: Towards real-time object detection with region proposal networks.* Paper presented at the Advances in neural information processing systems.

Sabour, S., Frosst, N., & Hinton, G. E. (2017). *Dynamic routing between capsules.* Paper presented at the Advances in Neural Information Processing Systems.

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). *Inception-v4, inception-resnet and the impact of residual connections on learning.* Paper presented at the AAAI.

Walach, E., & Wolf, L. (2016). Learning to count with CNN boosting. Paper presented at the European Conference on Computer Vision.