Traffic Anomalies Detection with A Semi-supervised Approach

Daniel Shumaker
Michigan State University
shumak37@msu.edu

Luan Nguyen
Michigan State University
nguye590@msu.edu

Pengyu Chu
Michigan State University
chupengy@msu.edu

Abstract

Anomaly detection is one of important fields in smart city. This helps automatically detect unusual actions such as lane violation, illegal U-turn, wrong driving, etc. In this paper, we proposed a fusion approach by combining both supervised and unsupervised learning to solve the anomaly detection problem. In specific, we develop models for car and road detection and tracking. Based on the relative position between cars, road, and cars' velocity to determine whether an anomaly happens or not. Experimental results shows that our approach achieves xx% accuracy for Track 3 AI city challenge.

1. Introduction

The opportunity for cities to use traffic cameras as city-wide sensors in optimizing flows and managing disruptions is immense. Where we are lacking is our ability to track vehicles over large areas that span multiple cameras at different intersections in all weather conditions. To achieve this goal, one has to address three distinct but closely related research problems: 1) Detection and tracking of targets within a single camera, known as multi-target single camera (MTSC) tracking; 2) Re-identification of targets across multiple cameras, known as ReID; and 3) Detection and tracking of targets across a network of cameras, known as multi-target multi-camera (MTMC) tracking. MTMC tracking can be regarded as the combination of MTSC tracking within cameras and image-based ReID with spatio-temporal information to connect target trajectories between cameras. Since the goal of the competition is to develop a new approach for better car detection and tracking as well as anomaly detection, we aim to propose a novel model for anomaly detection from highway video cameras.

2. Problem Description

Our project comes from *aicitychallenge.org*: a competition held every year that challenges programmers to expand our society's knowledge in the realm of self driving cars with a focus on urban car automation. Every year they offer new challenges. Our project is based on challenge number 3-Ťraffic Anomaly Detection.ˀ Like the name suggests, our challenge is to detect behavior that deviates from the way vehicles normally act. Examples of anomalies that we are looking for are illegal lane crossing, wrong direction driving, swerving cars, illegal U-turns, stopped vehicles, and crashes with a few stretch goals like detecting the lack of headlight usage and excessive breaking. Since the ability to detect those activities depends heavily on the data set quality, the developed model must be robust to a wide range of situations such as zooming, vibrations, and sudden direction changes. Most of the videos in the challenge are retrieved from highway cameras. Thus, a place where U-turns detection will be hard to train due to the infrequency of their occurrence in that setting.

This problem is important because it can help save the lives of people as cars more and more become autonomous. With sensors tracking traffic, signaling systems, and infrastructure, our transportation systems are becoming smarter. With computer vision and deep learning, there are many opportunities to solve real-world problems with data gained from multi-cameras. The major causes of accidents are the outliers to normal drivers. The drivers that go much slower on the free way, that aggressively change lanes, and ride others' bumpers are a major cause for sober accidents. Another major accident is caused by drivers under the influence of drugs. These drivers have a differing, and more dangerous set of driving patterns. Both groups can be extremely dangerous, particularly in traffic highways and intersections. With software that can detect anomalies in behavior, a computer could spot potential hazards and, depending on where it is implemented, could notify police in the area of a potentially drunk driver, or could tell an autonomous vehicle which cars to give a wide berth to. Solutions could potentially get the humans in the loop to pay attention to meaningful visual information in situations where timely intervention can save lives [7].

Unfortunately, progress has so far been limited for several reasons like missing data labels, and the lack of high-

quality models that convert data to decent forms. In other words, it is hard to get data with labels from the real world. Due to the lack of labels, anomalies cannot be classified with current algorithms. Thus, we plan to seek a semi-supervised approach to address this problem and focus on the research and development of techniques that rely more on transfer learning and semi-supervised learning.

By using pre-trained models like VGG 16 [9], we can leverage transfer learning to combine these models with our approach and detect the abnormal traffic behaviors from traffic camera video data which is provided by NVIDIA corporation.

3. Related works

There are not so much previous works in anomaly detection in highway videos. Giannakeris *et al*. [2] calculated speed of running cars to detect the differences in speed among different cars on the road to determine the unusual things. However, this way of calculating depends heavily on the accuracy of the car detection and tracking. If one of these terms are not correct, it can dramatically lead to wrong results. Zhang *et al*. [12] used deep neural network to summarize video and process anomaly. This approach seems to be stable to noise and zoomed senarios. However, it needs to process the whole video several times to summarize and analize the results. To fuse the results of different camera viewpoints, Tang *et al*. [10] ultilized tracking module to keep track each car on the road. They further estimate speed based on 3D features of visual and semantic meaning. Wei *et al*. [11] proposed an unsupervised approach that makes the model to be near real-time processing. The key idea of this approach is based on the background modeling which finds the differences between frames. All of the mentioned approaches are either easily sensitive to camera moving or insufficient capacity to process long video. Our approach, which is the fution of supervised and non-supervised learning to levera those defects with decent results.

4. Dataset

Our solution to the problem described above begins with the data set we have to work with. *AI City Challenge* provided us with the data for this challenge. The training and test set are visually pretty similar. Each folder contains 100 mp4 files. Each video is about 15 minutes long, usually is set in a highway, and, not surprisingly, has a lot of cars in it. In the README file for the training videos, the challengers give us the areas they believe to have issues. Surprisingly, most videos in these sets don't have any anomalies in them, or at least were not recorded in the README document.

By going in depth of the dataset, it is very likely that there will be wind, and the video will need to be stabilized before processing. The cameras used to record the videos

Figure 1. Image from video 17

Figure 2. Image from video 41

Figure 3. Image from video 66

differ in quality greatly. There are many factors that vary in different videos such as weather condition, time of the day filming took place, geographical location, and the andlge of the cameras. Another observation we made after looking at the videos and at the suggested anomalies of the training videos was a lack of diversity in the errors. Much of the errors that occur are vehicles stopped on the sides of the road, and many of the errors that are not about stalled vehicles, do not appear to have any errors at all.

To get a good result and make models be stable, we apply preprocessing techniques to normalize the data. Figure **??** shows a few images from the videos in the data set. An example of an image with an anomaly in it is presented in Figure 4.

The challengers require that we report our anomalies in the same way that they do: give the video name, the start

Figure 4. Image from video 2. Contains Anomalies. In the small red circles, two cars are stopped on the side of the road.

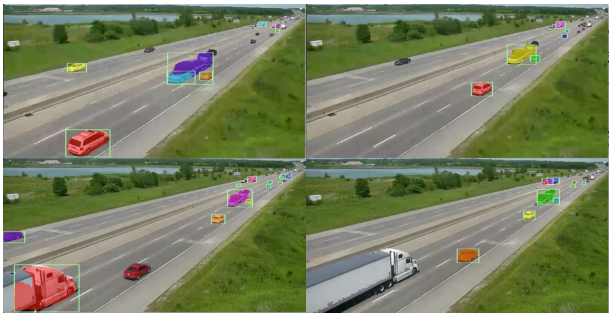

Figure 5. Detect cars through videos with Mask R-CNN. The sequence is from left to right and top to bottom.

time of the anomaly, and the end time of it in seconds. If the anomaly continues past the end of the video, we are to give the last second of the video as the stopping time.

5. Proposed method

Based on the definition of anomalies, we mainly have two types of anomalies, which are running off the road and stalling on the highway. Following criteria are the guide that we used to develop our model:

- Detect every cars or trucks in the videos and point out their locations;
- Track these vehicles in the sequence of each video;
- Measure the speed of cars based on a continues frames in the videos and judge its status;
- Detect the road area to check whether a car runs off the road or not.

In this section, we introduce our methods mentioned in details.

5.1. Vehicle Detection

In the realm of objective detection, we have lots of state-of-the-art methods including two-stage detection methods like Faster R-CNN [9] or one-stage detection represented by YOLOv3[8] and SSD[6]. After a comparison among these methods, we choose the Mask R-CNN as our detector because it has the best accuracy over the benchmark.[3].

To increase the stability of the model, we use state-of-the-art two stages objects detection and semantic segmentation CNN, Mask R-CNN [3] with ResNet101 as the backend, to finish this work. In addition, for improving the detection accuracy, we annotated 498 images extracted from these videos to train our own model, which is just for car detection.

5.1.1 Pre-trained Mask R-CNN and Applied in Videos

Because it is hard to train a CNN model from scratch, so we leverage the pre-trained model in COCO dataset[5], which is a public images dataset, to detect cars in our scene. On the other hand, Mask R-CNN is a classifier based on images not videos, so we needed to modify it to work with our given data set.

We used a video library named 'moviepy' to deal with this problem. After extracting a sequence of frames from the video, we applied the detector on each frame. The details will be displayed in the code. Figure 5 displays an example of a continuous result.

Finally, we output our detections as a text file for each video consisting of labels and bounded boxes with coordinates.

5.2. Vehicle Tracking

During the vehicle tracking, we leverage a relatively mature method, Simple Online and Realtime Tracking (SORT)[1], after considering the realtime performance we need. We use the SORT algorithm to calculate the similarity among different patches in the sequent two frames from the last module's output. Then we build a connection for each car from different cars, which means we can find a temporal relationship for each car. The basis of the algorithm is that it compares a frame with the next one among the coordinates and size of bounding boxes , and then compute a velocity vector. More specifically, it uses the following flows to process the calculation:

- It uses Kalman filters to compute the velocity factor. A Kalman filter essentially does some math to smooth the velocity and direction computation by comparing the predicted state and the real detection given by Faster R-CNN. Obviously, we changed it to a better model Mask R-CNN;
- It uses an assignment cost matrix that is computed as the intersection over union (IOU) distance between each detection and all predicted bounding boxes from

the existing targets (which is putting all the dimensions in a normalized matrix). Then the best match is computed using the Hungarian Algorithm, which is a way to fast compute lots of matrices;

- It also deals with the score of the detections, a quality other tracking methods don't use. This quality of the algorithm allows the tracker to choose between two close detections based on the better score.

Compared to the last section output, the tracker annotates the identity for each car. Based on the example, we can observe id=1885 and id=1886 are moving in these five frames and the id=1887 disappears after the third frame. We can exploit the moving information to verify if the car stalls or not and whether it's included in the road area or not.

5.3. Anomaly detection based on histogram of pixels

This way of approach is unsupervised learning. The key ideas are to divide the frame video into small regions having the size of 32×32. We calculate the change in pixel intensity in each small region to create the histogram of in temporal features. This method is based on the assumption that if one anomaly happens, the histogram inside that region will be different from the rest of time frames. Although this approach is not generalize enough to cover all situations, it has decent results in most of cases.

5.4. Anomaly calculation

In this section, we demonstrate the anomaly calculation using cars and road relative position as well as the speed of each car. Since the tracking and detection results are not always correct, we leverage each car's id to calculate their speed fluctuations and stopping duration, in order to judge whether this is an anomaly event or not. We generate happening time as the output and compare it with the ground truth to measure our algorithm's accuracy. Algorithm 1 shows the overall anomaly calculation in our approach.

6. Experimental results

6.1. Speed Validation

After vehicle tracking, we have a sequence of each car that we've tracked in the last section. We assume the sequence is true (since we still face some tracking errors but we'll address it with a direction limitation in the next work). Since the camera position and perspective won't change, we can use the motion of the target centroid and the gap between frames in the videos to calculate the speed.

$$P(x, y) = \frac{(Box_{11} + Box_{12} + Box_{11} + Box_{22})}{4}$$

Algorithm 1 Anomaly detection using relative position and speed

Input: Bounding boxes of cars, road and tracking
Output: Anomaly time frame in the video

1: Convert the data into center position and speed form; [id, time, center, speed];
2: Subtract adjacent speed and position, and get the difference of positions and speed: DOP and DOS;
3: Set threshold for DOP and DOS and for ones lower that DOP and DOS, we set up them as 0;
4: Discard stationary objects(that could be a wrong detection) and discard unstable tracking, which means some trackings occur for a short time.
5: Discard the running cars during all the time and the stopping cars due to the traffic lights through a stopping time ratio threshold;
6: Merge anomalies at the same or close position. We think these cars crash due to the same reason;
7: Output the final anomalies .

$$D(t) = \Sigma_{i=0}^{k} \frac{P_{t-i}}{k+1}$$

$$V(t) = \frac{D(t)}{gap}$$

$$h(x) = \begin{cases} 1, & \text{if } V(t) > threshold \\ 0, & \text{otherwise} \end{cases}$$

$P(x, y)$ is the representation of a target centroid, $D(t)$ is the average moving distance, and $V(t)$ represents the speed of a pixel's motion with the gap depending on the extracting rate from videos. The $h(x)$ classifies the anomalies and threshold based on the training results. So from this equation $h(x)$, we can know if a car stalls or not in the time t

6.2. Road Detection

Our cameras are all fixed and then that means the road will be static as the background. So we can use a simple algorithm Canny Edge Detection and Hough Transform to estimate the road areas in the each video. That's our subsequent work to finish.

After that, we'll compare whether the centroid of a car is in the area or not and then validate whether it ran off the road or not, which is the other type of anomaly we foresee needing to deal with.

$$P(x, y) = \frac{(Box_{11} + Box_{12} + Box_{11} + Box_{22})}{4}$$

$$f(x) = \begin{cases} 1, & \text{if } x \in Region_{road} \\ 0, & \text{otherwise} \end{cases}$$

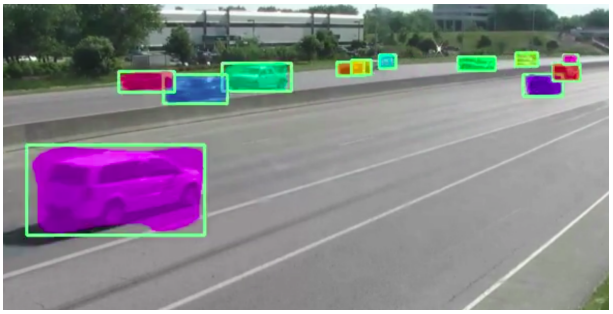

Figure 6. Cars detection based on fine-tuning Mask-RCNN.

Figure 7. Road detection based on relative fusion of cars position in frames.

Figure 8. Some difficult situations in recored videos: different illuminations, viewpoints, errors between video frames, etc.

In this equation, P(x,y) represents the coordinates of the centroid of each car and $Region_{road}$ can be depicted with 0-1 Matrix. So the $f(P(x,y))$ will express whether it's an anomaly or not.

6.3. Results and analysis

6.3.1 Qualitative

The final goal of the model is to ouput the anomaly frame, all objects (cars) in each video frame need to be detected and tracked correctly. Figure 6 and 7 show some examples of our cars and road detection results.

The difficulties in the data is not only about zoom or shaky problems, illumination, different camera viewpoints. Figure 8 shows some of difficult situations in the video with different scenarios.

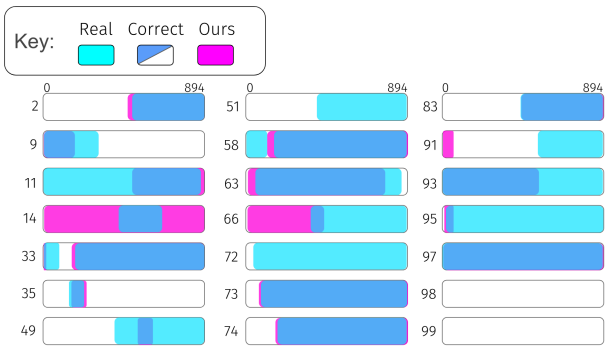

Figure 9. Interactive quantitative comparison between our proposed model and validation set.

6.3.2 Quantitative

We used F1-score to evaluate the correctness of the model for anomaly detection performance and detection time error, measured by RMSE. Specifically, the final score will be computed as $S_{Final} = F1 \times (1 - NRMSE)$, where $F1$ is the F1-score and $NRMSE$ is the normalized root mean square error (RMSE). The score ranges between 0 and 1, and the higher scores reveal the better model.

For the purpose of computing the F1-score, a true-positive (TP) detection will be considered as the predicted anomaly within 10 seconds of the true anomaly (i.e., seconds before or after) that has the highest confidence score. Each predicted anomaly will only be a TP for one true anomaly. A false-positive (FP) is a predicted anomaly that is not a TP for some anomaly. Finally, a false-negative (FN) is a true anomaly that was not predicted.

We compute the detection time error as the RMSE of the ground truth anomaly time and predicted anomaly time for all TP predictions. Normalization will be done using min-max normalization with a minimum value of 0 and a maximum value of 300, which represents a reasonable range of RMSE values for the task.

Figure 9 shows the quantitative results interactively of our model on validation set.

7. Conclusion and future Work

We have presented the model and architecture for the anomaly detection. The proposed approach contains two main modules: cars/road detection and cars tracking with speed estimation. Experimental results show that our approach can get correct results in almost videos which small differences compared to groundtruth. We believe that our approach method can be used as a based line for anomaly detection development in the future. Further improvements can be exploited in following ways: improving tracking method to handle different frame changes, increasing the speed of detection module to make it faster for real-time process.

References

[1] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468. IEEE, 2016.

[2] P. Giannakeris, V. Kaltsa, K. Avgerinakis, A. Briassouli, S. Vrochidis, and I. Kompatsiaris. Speed estimation and abnormality detection from surveillance cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 93–99, 2018.

[3] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[4] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[6] M. Naphade, D. C. Anastasiu, A. Sharma, V. Jagrlamudi, H. Jeon, K. Liu, M.-C. Chang, S. Lyu, and Z. Gao. The nvidia ai city challenge. *IEEE Smart-World, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (Smart-World/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, 2017.

[7] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[8] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[9] Z. Tang, G. Wang, H. Xiao, A. Zheng, and J.-N. Hwang. Single-camera and inter-camera vehicle tracking and 3d speed estimation based on fusion of visual and semantic features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 108–115, 2018.

[10] J. Wei, J. Zhao, Y. Zhao, and Z. Zhao. Unsupervised anomaly detection for traffic surveillance based on background modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 129–136, 2018.

[11] Y. Zhang, M. Kampffmeyer, X. Zhao, and M. Tan. Deep reinforcement learning for query-conditioned video summarization. *Applied Sciences*, 9(4):750, 2019.