Front-end Collision Prevention

By Xi Chen, Thomas Huang, Pablo Moncada, and Charlie Rogers

1. Problem Definition

According to National Highway Traffic Safety Administration (NHTSA), in 2012, one-third of all crashes on road are caused by front-end collisions [1]. Front-end collisions are hard to avoid simply by the effort of the driver, since it always involves careless driving of others. The two most common situations are free lane changing of the cars beside you and sudden braking of cars in front of you. This can sometimes be difficult to spot, especially if you have been on the road for a long time, and the consequences could be drastic. Our team is setting out to help drivers detect if there is a potential of a front-end collision on the highway using any camera, such as a phone camera, on a dashboard.

Our team constrained the problem by limiting the input to forward-facing dash camera feeds with straight roads, daytime, and consistent light conditions to reduce the complexity of the problem.

2. Methods

Both situations where front-end collisions occur mentioned above involve the behavior of slamming on the brake of the car in front, so we need to send an alert to the driver as soon as the car in front gets closer and the brake light illuminates. Our detector is therefore split into two components: a car bounding box detector and a brake light detector.

2.1 Car Bounding Box Detector

The car bounding box detector detects the presence of cars in the image, and returns a bounding box indicating the location of the detected car. For this detector, we used a method called Single Shot MultiBox Detector (SSD) [2]. SSD uses a single deep neural network, which makes it easy to train and run. This network is shown below in Figure 1.

In this paper, we will explore two features of SSD that makes it perform well: multi-scale feature maps and default boxes. Multi-scale feature maps refers to the use of multiple feature maps at different sizes. This allows us to perform detection at multiple different scales, which allows us to handle objects at various sizes. Default boxes refers to the use of bounding boxes at fixed locations with fixed aspect ratios. Detection is performed on each of the default boxes, and the box with the highest detection score is returned as the resulting bounding box. The use of the default boxes is shown to improve performance.

For this paper, we use pre-trained weights on the VOC2007 dataset due to time constraints.

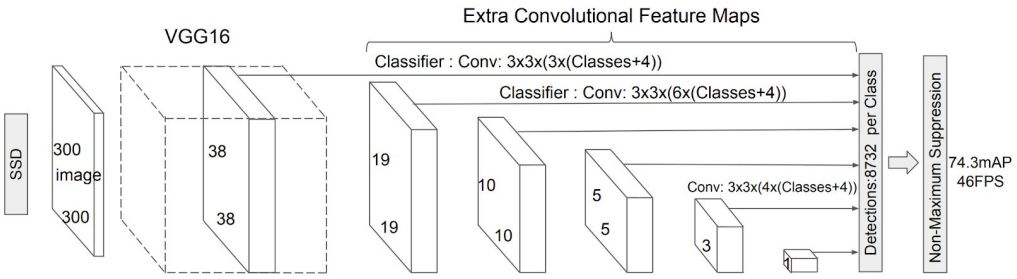

Figure 1: Architecture of SSD

2.2 Brake Light Detector

We based our brake detection strategy off of the method used by Tantalo and Merat in their paper Brake Light Detection by Image Segmentation [3]. First, we convert the RGB input image to the HSL color space so that we have a more precise way of differentiating between illuminated and non-illuminated brake lights. Next, we shift the hue by 180° and segment out a binary image that represents the regions with high luminance and reddish hue. Finally, we perform a morphological closure (erosion of a dilation) on the binary image to remove small holes.

After this image processing, we run a threshold and contour-based OpenCV blob detection on the binary image. To ignore noise, we then filter every pair of detected blobs using four rules that represent general properties of pairs of brake lights. First, we make sure that both blobs have similar areas. Then, we confirm that the blobs are an appropriate distance apart relative to their size and that the blobs are aligned approximately horizontally. Finally, we verify that one blob is in each half of the vehicle bounding box for each pair of blobs.

2.3 Analysis Pipeline

Given a group of potential cars from our car bounding box detector we must first determine the car that is directly in front of our driver. We assume that the camera is installed properly so it will be centered on our current lane. From that assumption we choose the car whose center (x,y) coordinates minimizes this cost function:

$$C(x,y) = (y - Im[y].size())^2 - \eta(x - 0.5 * Im[x].size())^2 ,$$

where η is a hyperparameter that biases towards cars in the center of the image rather than further from the bottom (in our implementation we chose $\eta = 10$).

Once we determine the relevant bounding box, we mask the image with the selected bounding box and run our brake light detection function. If the brakes are currently activated we check if

the scale of the car has increased over the last second. This is done by performing a linear regression on the area of the last 5 bounding boxes associated with that car and determining if the slope of the boxes has increased more than α, in our implementation we chose $\alpha = 0$.

There is a tricky balance between false positives and true negatives, therefore we add a second layer of logic that includes a transition stage from "clear" to "alert." We check the last 5 states of our program, and if 3 out of the last 5 states have been in alert, we notify the driver. This means that the reaction time of our program is at best 0.6 seconds since we calculate a new bounding box every 0.2 seconds.

3. Results and Limits

3.1 Functionality

As shown in Figure 2 below, the three frames are in sequence and they capture the braking motion of the car in front. As the bounding box getting larger in the frame and the brake light illuminating, the car brakes are detected and alerts are sent to drivers. The algorithm functions well in most of the test video. However, the algorithm fails in a few cases (Figure 3): low quality of video, sudden appearance of target, and exhaust fumes or dust which block brake lights.

Figure 2: Resulting detections on 3 frames of a video. Left: no brakes detected. Middle: possible brake detected. Right: brake detected.

Figure 3: Resulting detections on 3 frames of a video. Left: no brakes detected. Middle: sudden appearance of true target and wrong target detected. Right: target detected but too late.

3.2 Real-time

The algorithm can process 1.23 frames per second on a 2.9 GHz Intel Core i7 Processor and Intel HD Graphics 630 1536 MB GPU, which means a 0.81 seconds lag. This 0.81 seconds lag does not take into consideration of the latency of taking photos and the processing power difference, but it can already lead to fatal damage on the highway. Improvement is needed for the algorithm to be applicable. We provided some possible improvements in the next section.

4. Future Work

If we were running our program on a more powerful computer, we could calculate a new bounding box more often, resulting in a better reaction time for our program. Also, the model we used for our car bounding box detection was originally trained on many different objects. Training the model specifically for cars might improve performance. Furthermore, if we had access to accelerometer data we could determine if the driver was already braking, adding additional logic to prevent annoying or redundant notifications to the driver.

5. Conclusion

Under the specified constraints, our method of detecting the potential of a front-end collision performs reasonably well, although it clocks in at slightly less than real-time processing speed as mentioned above. However, with some calculated improvements, our system could prove to be a valuable safety measure for millions of drivers.

6. References

1. NHTSA. (2017). *Safety Technologies*. [online] Available at: https://www.nhtsa.gov/equipment/safety-technologies [Accessed 7 Dec. 2017].

2. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. and Berg, A. (2016). SSD: Single Shot MultiBox Detector. *Computer Vision – ECCV 2016*, pp.21-37.

3. Tantalo, J. and Merat, F. (n.d.). Brake Light Detection by Image Segmentation. [online] Pdfs.semanticscholar.org. Available at: https://pdfs.semanticscholar.org/0441/572cfe721dcdb3c8e5ab94cdae9883e77085.pdf [Accessed 7 Dec. 2017].