Vehicle Speed Detection and Safety Distance Estimation Using Aerial Images of Brazilian Highways

Mateus Eloi da Silva Bastos[1], **Vitor Yeso Fidelis Freitas**[2],

Richardson Santiago Teles de Menezes[1], **Helton Maia**[2]

[1] Department of Computer Engineering and Automation

UFRN, Natal-RN, Brazil

[2]School of Science and Technology

UFRN, Natal-RN, Brazil

helton.maia@ect.ufrn.br

Abstract. *In this study, the computational development conducted was based on Convolutional Neural Networks (CNNs) and the You Only Look Once (YOLO) algorithm to detect vehicles from aerial images and calculate the safe distance between them on Brazilian highways. Data analysis was performed in a dataset composed of 896 images, recorded on videos by a DJI Spark Drone. Thus, with this dataset, 60% of the images were used for training, 20% for validation and the remaining 20% for tests. Experiments were conducted to detect vehicles in different network architecture configurations, the best result was obtained with the YOLO Full-608, reaching an Average Accuracy (mAP) of 95.6%. The accuracy of the results encourages the development of systems capable of estimating the safe distance between vehicles during their movement on the highways, allowing to detect and minimize the risk of accidents.*

1. Introduction

Traffic accidents are certainly one of the most worrying problems of modern life, the World Health Organization (WHO) reveals that annually about 1.35 million people have their lives disrupted due to a traffic accident [Organization 2008]. Also, between 20 and 50 million people suffer nonfatal injuries. The constant increase in the number of vehicles in Brazil and especially in the northeast region brings severe problems of urban mobility, such as accidents, traffic jams, and structural issues in highways [Júnior and da Silva 2019].

The northeastern region of Brazil has about 16,999,050 vehicles occupying the third position in the list of regions with the most vehicles. According to data published by the federal highway police, speeding, and the short distance between vehicles constitute two of the main reasons for accidents caused by human factors [PRF 2019]. Interventions by the authorities to alleviate these problems are often old and inefficient. Today, much research has been done to apply machine learning and computer vision to solve many

problems related to urban mobility, traffic density, and road accidents [Zear et al. 2016] [Mandhare et al. 2018] [Buch et al. 2011].

Researchers at the Alan Turing Institute in London and the Toyota Mobility Foundation in Japan recently launched a joint project aimed at improving traffic management systems through the use of artificial intelligence [toy 2019]. Through the images acquired by the monitoring cameras, processed and analyzed by various Machine Learning (ML) algorithms, it is possible to obtain important information such as the individual speed of vehicles, the distance between them, traffic density, and even the identification of the type of vehicle that is traveling. With the acquisition of information on the highways, it is possible to create computational solutions for efficient road monitoring with the main purpose of reducing accidents [Blosseville et al. 1989].

Currently, the use of Unmanned Aerial Vehicles (UAVs) to solve traffic-related problems is growing rapidly. Features such as flexibility, mobility, and good resolution in image capture, make UAVs one of the best choices for road monitoring [Koubaa and Qureshi 2018]. With the growth of deep learning-based algorithms, especially Convolutional Neural Networks (CNNs), when combined with the extensive production [Dro] and application variety of UAVs, it provides an ideal scenario for the growth of research and development of new applications related to this area.

By combining the advantages of CNNs in classification and object detection problems with the evolution of Graphics Processing Units (GPUs), it is possible to solve problems related to real-time image capture and processing [M. H. Putra]. In this context, this paper considers the scenario of traffic monitoring from aerial images using an UAV. Initially the detection and classification of vehicles on the highways is performed. Then, the distance between the moving vehicles is calculated, and it is possible to estimate whether they are traveling safely or not, according to Brazilian law.

2. Methods

The C / C ++ programming language and the YOLOv3 algorithm [Redmon and Farhadi 2018] were used to detect vehicles on the highways. We built a dataset with aerial images, captured in high definition by a Drone model DJI Spark. The computational development was performed on a computer equipped with a CPU AMD Athlon II, 4GB of RAM, GPU NVIDIA GeForce GTX 1070, OS Ubuntu 18.04 LTS, CUDA 9.1 and CuDNN 7.1. The structure of this work and all development steps are summarized in the UML-based activity diagram shown in Figure 1.

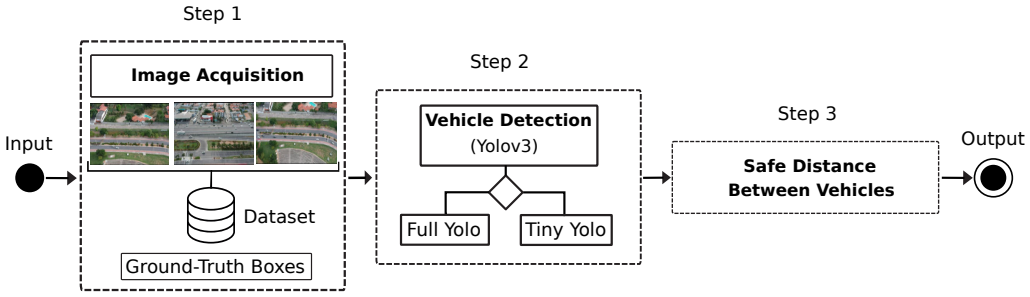

Figure 1. UML activity diagram showing all the steps of computational development.

In step 1, Initially, the aerial images of the traffic on the highways are captured and recorded on video using the drone. After that, a sample of the images is selected for a manual selection of the ground-truth boxes enclosing the objects of interest, which will compose a database to be used in the following steps.

In step 2, the training and validation of the networks are performed. We used the YOLOv3 algorithm with different architectures, in Full and Tinny [Adarsh et al. 2020] versions. Vehicle detection and classification is also performed in the test images in order to evaluate the trained models.

In step 3, considering the previous information on the detection and classification of vehicles, the safe distance between then is calculated, and it is possible to display the result in real-time.

2.1. Dataset

The images recorded for this work were acquired in Full High Definition (FHD) with resolution of 1920×1080 pixels using a DJI Spark drone. Examples of this acquisition can be seen in Figure 2, where images of highways are displayed, close to the Federal University of Rio Grande do Norte (UFRN).

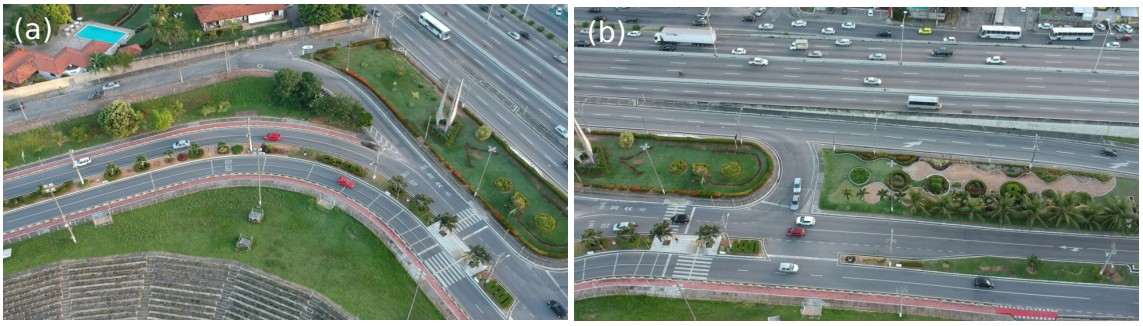

Figure 2. Images captured by DJI Spark drone near UFRN (a) Coronel João Medeiros Street, at 100m altitude. (b) Highway BR-101 at an altitude of 100m.

The videos were recorded at a rate of $30fps$, and then a sample of 896 frames spaced in time was selected, avoiding similar images. The images of the dataset built for this work were randomly divided into three parts, they are: 60% for the training set, which is equal to 536 images, 20% of the images for validation, which is equal to 180 images, and finally, 20% for the test set with 180 images. In our dataset, for each image it is possible to select several objects from different classes. It is also possible to notice a higher occurrence of cars than motorcycles or buses. Table 1 shows for each class of the dataset, the number of occurrences of the respective objects.

Table 1. Number of objects for each class in the dataset.

Class	Training	Validation	Test
Car	8982	2888	3376
Bus	303	109	112
Motorcycle	841	268	339

2.2. Image annotation

Image annotation is an important step for object detection using CNNs and supervised learning. The image annotation process consists of saving the positioning of the objects of interest for each image in the dataset. The relationship between each image and its respective positioning is essential information for learning the neural network.

In addition, positions should be properly normalized with x and y coordinates between 0 and 1, where $x = 0$ and $y = 0$ is the upper-left edge of the image, and $x = 1$ and $y = 1$ is the lower right edge of the image.

In this way, it is possible to correctly send the spatial information of the objects to the neural network and its respective architecture. The software YoloMark and LabelMe [Wada 2016] were used to make these notes, an example of their operation is shown in Figure 3.

Figure 3. Example of how LabelMe works marking vehicles in the image.

Table 2 shows an example of the annotation for the data associated with Figure 3, representing the parameters of the object's bounding box and its respective class information. Subsequently, the data are transformed into tensor labels to be used in the training step of the neural network.

Table 2. Data annotation pattern for image. [3]

Class	X Center	Y Center	Width	Height
0	0.2850	0.3250	0.3245	0.2055
2	0.8245	0.5277	0.1754	0.1444

2.3. Convolutional Neural Networks

Currently, the most efficient algorithms for image classification and object detection use deep learning architectures, with many specialized layers to automate the process of filtering and extracting resources [Menezes et al. 2019]. The ability to learn from the extraction of many features is one of the most relevant utilities present in algorithms based on deep learning using CNNs.

The component for extracting features is the convolutional layer at the input of the neural network, which uses a very common image processing operation, called two-dimensional discrete convolution between $m \times m$ kernel and two-dimensional data, therefore, convolution can be seen as a specialized type of linear operation performing the role of a filter to a determined input [Goodfellow et al. 2016].

Computational cost and irrelevant features are frequently inconvenient problems in this approach. Therefore spatial dimension reducer layers like polling layers are commonly used after convolutional layers.

2.4. Object Detection and YOLOv3

The complexity involved in the object detection problem entails a higher computational cost for the overall approach. In this work, the prediction of the spatial location of the objects is showed by a bounding box around the objects of interest, i.e car, bus, and motorcycle. These bounding boxes are shaped according to the detected object with additional class confidence associated.

The third version of the You Only Look Once (YOLO) [Redmon and Farhadi 2018] object detection algorithm was selected for this work due to its accuracy and real-time processing capacity. It differs from other object detectors by using a single CNN for both classification and localization of the objects.

To perform object detection, the YOLO network first divides the input image into a $N \times N$ grid cells, and each cell is responsible for detecting any object's centroid inside of it. For this, logistic regression is used to calculate the confidence score of an object in each bounding box.

The analysis of a frame in the YOLO network consists of three steps. First, the input image is resized, then a single CNN is run on it, and in the last step, thresholds using non-maximum suppression are applied in the resulting detections as described in [Redmon et al. 2016]. Figure 4 depicts how an image of the dataset of this work is processed.

In addition to the full version, a smaller version of YOLO called tiny YOLOv3 was also used in this work. This reduced version has a smaller feature extractor, reducing the computational cost, and in some cases, maintaining the accuracy of object detection.

The feature extractors of the two versions trained for this work were initialized using pre-trained weights trained on the ImageNet dataset [Deng et al. 2009]. This transfer learning approach saves a considerable amount of training time if compared to train using random initial weights or some probability distribution.

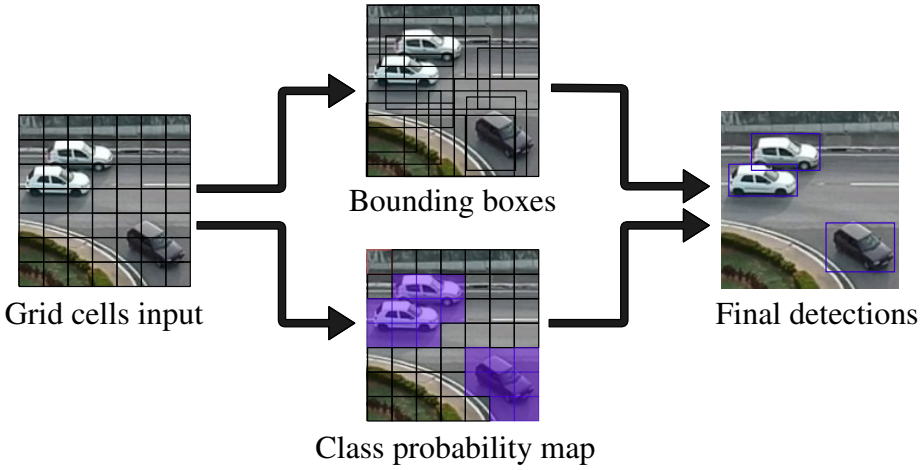

Figure 4. Yolo object detection approach.

2.5. Safe distance between moving vehicles

Drivers are often faced with a scenario where they must follow other cars on the road, in these situations is crucial to keep what is called a *safe distance* to the car in front.

Safe distances obey the laws of each country and may vary according to the speed of the vehicle, driving conditions, and climate, types of vehicles, etc. Following another vehicle very closely is known as tailgating, which, if carried out without proper precautions, can lead to collisions, which is the most common type of problem [Lee et al. 2007].

In this context, a tool is proposed to automatically detect the safe distance between vehicles, where, using the object detector based on YOLOv3 and the Python programming language, a fully automated way to calculate these distances is achieved. Figure 5 illustrates the calculation of the safe distance between vehicles.

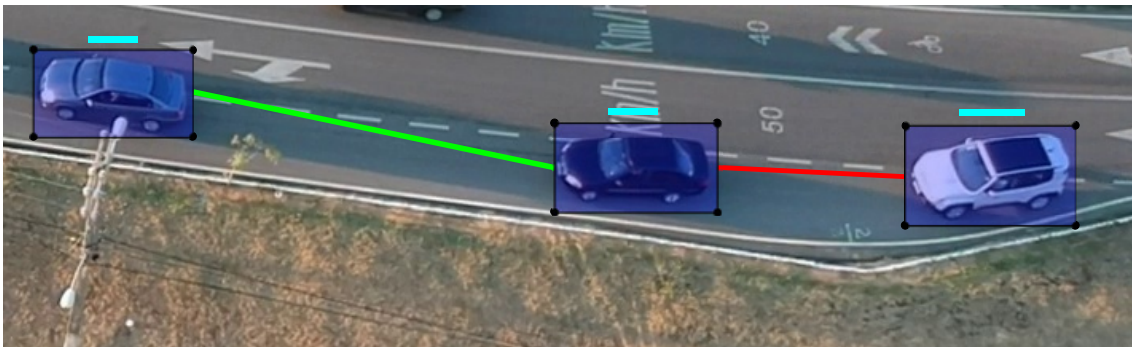

Figure 5. Illustration about the safe distance between vehicles. The tiny bars on the top of vehicle boxes are proportional to the vehicle's velocity. The bigger bars represent the distance between vehicles, which the color says if the distance it's safe or not.

3. Results

3.1. Vehicle tracking

The models were evaluated using the Mean Average Precision (mAP), which is the mean of Average Precision (AP) for the classes. AP is the area under the interpolated precision-

recall curve. Precision and recall were calculated using an Intersection Over Union (IOU) threshold of 0.5, delimiting true and false predictions.

All the networks are trained using a batch of 64 images per iteration, a momentum of 0.9, and decay of 5×10^{-4}. Many approaches to data augmentation were used in the training set, in particular, position augmentation such as scaling, rotation, flipping, cropping, translation, and padding, as well as color augmentation like brightness, saturation, contrast, and hue.

Two sizes of YOLOv3 models have been trained, they are: (416 x 416) and (608 x 608) for network input. Higher size models make better generalizations when you have a large volume of data, despite having more computational cost. Loss and mAP curves can be seen in Figure 6. YOLOv3-608 detected our test images very well, getting an mAP of 95.6%. With that, reliable predictions can be used for our safety distance estimator.

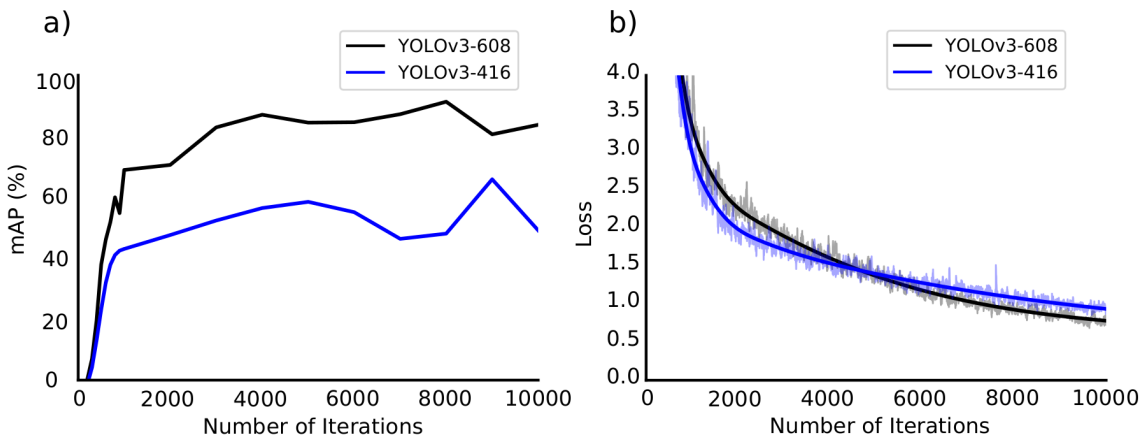

Figure 6. a) YOLOv3 validation mAP. b) YOLOv3 training loss.

The tiny YOLOv3 is an alternative to limited hardware, due to a good trade-off between inference time and accuracy. In the dataset used, which has a variety of sizes for some classes, the 2-scale model did not perform well in our experiments. In this way, it was necessary to add one more scale of prediction in Tiny YOLOv3, similarly to YOLOv3. Therefore, it is expected that the model will better generalize objects of different sizes and in regions with a higher density of objects. We did this for two models resolutions, (416 x 416) pixels and (608 x 608) pixels.

This idea was confirmed by the experiments. Test mAP increases 28.8% between Tiny YOLOv3 with 2 scales and Tiny YOLOv3 with 3 scales. The mAP and loss curves can be seen in Figure 7. In general, our tiny network performed well on test set and achieved a good 40.1 fps running in real-time. Computational costs for (608 x 608) resolution are significantly increased, therefore decreasing fps in a half. These relations are shown in Table 3.

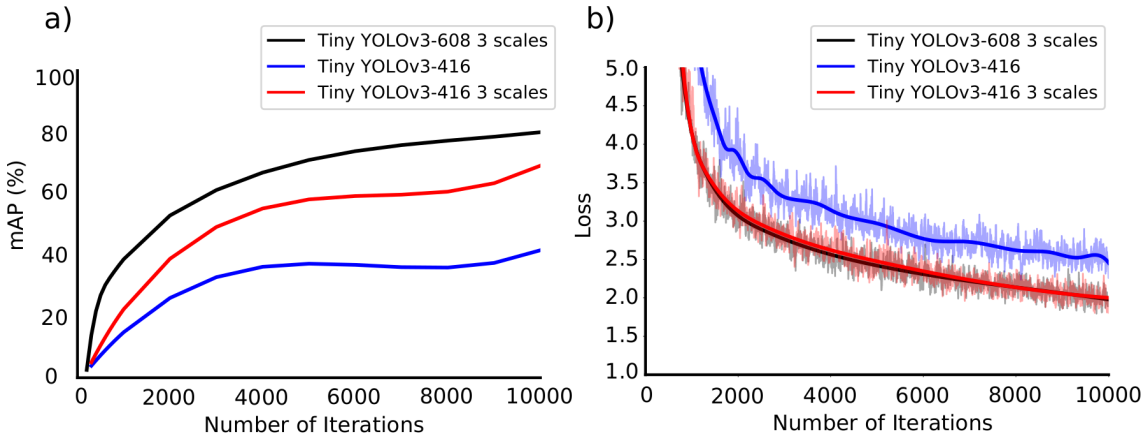

Figure 7. a) YOLOv3 Tiny validation mAP. b) YOLOv3 Tiny training loss.

Table 3. Comparisons of the networks.

Network	FPS	mAP (%)	Car	Bus	Motorcycle
Tiny YOLOv3-416	**92.8**	44.7	44.1	72.3	15.8
Tiny YOLOv3-416 3 scales	81.1	73.5	81.3	88.2	51.1
Tiny YOLOv3-608 3 scales	40.9	79.7	87.4	91.6	60.1
YOLOv3-416	32.1	67.7	77.6	92.0	33.7
YOLOv3-608	17.0	**95.6**	**96.6**	**98.4**	**91.8**

3.2. Safe distance tool

In the tool developed and shown in Figure 8, the user initially chooses the video where the detections will occur and then defines a region of interest (ROI) in the image, this will prevent unwanted detections from being made. This ROI is shown as the magenta rectangle in Figure 8.

The frames captured from the user provided video are then fed into the YOLOv3 algorithm, where the vehicles detection is done. With YOLO detections, the developed algorithm finds the center of mass for each bounding box and calculates the Euclidean distance to the nearest car.

The detected class, index and distance to the closest detected vehicle are respectively shown above each bounding box and by the side the vehicle velocity is displayed, as seen in Figure 8. To avoid the distance being calculated between vehicles on opposite sides of the street, the vehicles are considered to be in the same lane and share very similar values for the y-coordinate of the center of mass, so this problem is solved by imposing a limit for y-coordinate.

The distance calculated with the tool here proposed is in pixels, so to be translated to real-world measurements a series of calibrations steps must be taken [Lu et al. 2007]. With the proper calibration of the software, a safe distance is then defined, then any time a vehicle in the video is in a smaller distance the bounding box and the line to the closest vehicle are going to turn red, alerting the user of the infringement.

Figure 8. User interface developed for the proposed safe distance tool.

Speed detections are based on the comparison of positions frame by frame between objects predicted by YOLOv3. Each object position in frame f_i is compared to each object position in frame f_{i+1}, and the closest objects are understood as the same vehicle. After this, we count the pixels difference between the two positions of each vehicle, thus achieving a speed of pixels per frame, considering that drone camera approximately record videos at 30 fps.

Adding these distances per frame for each vehicle, and adding a Δt parameter, we can obtain an average speed. Using the calculation of the speed and the distance between the vehicles, it is possible to safely assess whether they are obeying the safe distance recommendations for traffic on the highway and also, to predict the probability of accidents.

4. Conclusion

In this article, a computational approach was presented that uses digital image processing and CNNs to detect vehicles in aerial images captured by a drone. Also, it was possible to calculate the speed of the vehicles and measure the distance between them, making it possible to verify that the vehicles maintain the safe distance necessary for traffic on the highway.

For the detection and classification, we made an experimental comparison between some configurations of Yolo CNN, changing the number of detection scales and the size of the network input, in order to obtain a good accuracy with less processing time. Finally, it was obtained a 95.6% of mAP with YOLOv3 and good trade-off between mAP 73.5% and fps 81.1 with Tiny YOLOv3.

With the safe distance tool where proposed traffic authorities around the world can have a reliable and fully automated way to track vehicle speeds and their distance to others, given that the tool can be easily ported to other scenarios by simply retraining

the YOLO classifier. Therefore hundreds of hours in human labor can be saved and the security and driving conditions in highways increased.

All computational tools develop during this work can be found in our repository available at `https://https://github.com/heltonmaia/ECT-proj-cnn-vant`, which, includes a tutorial for the usage of the tools proposed.

Acknowledgment

This work was supported by the School of Sciences and Technology at the Federal University of Rio Grande do Norte (ECT-UFRN).

References

Drone technology uses and applications for commercial, industrial and military drones in 2020 and the future.

(2019). Alan turing institute and the toyota mobility foundation collaborate on improving city planning and traffic management with artificial intelligence.

(2019). Avaliação das políticas públicas de transportes.

Adarsh, P., Rathi, P., and Kumar, M. (2020). Yolo v3-tiny: Object detection and recognition using one stage improved model. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 687–694.

Blosseville, J., Krafft, C., Lenoir, F., Motyka, V., and Beucher, S. (1989). Titan: A traffic measurement system using image processing techniques. In *Second International Conference on Road Traffic Monitoring, 1989.*, pages 84–88. IET.

Buch, N., Velastin, S. A., and Orwell, J. (2011). A review of computer vision techniques for the analysis of urban traffic. *IEEE Transactions on Intelligent Transportation Systems*, 12(3):920–939.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). Deep learning. vol. 1.

Júnior, C. and da Silva, V. (2019). Fatores associados aos acidentes de trânsito graves envolvendo condutores de automóvel e motocicleta: uma análise para as br 101, 116 e 230 na região nordeste em 2007 e 2016. Master's thesis, UFRN.

Koubaa, A. and Qureshi, B. (2018). Dronetrack: Cloud-based real-time object tracking using unmanned aerial vehicles over the internet. *IEEE Access:10.1109/ACCESS.2018.2811762*.

Lee, S. E., Llaneras, E., Klauer, S., and Sudweeks, J. (2007). Analyses of rear-end crashes and near-crashes in the 100-car naturalistic driving study to support rear-signaling countermeasure development. *DOT HS*, 810:846.

Lu, M.-C., Hsu, C.-C., Lu, Y. Y., et al. (2007). Improvements and application of the image-based distance measuring system. In *Proc. WSEAS Int. Conf.(CISST)*, pages 17–19.

M. H. Putra, Z. M. Yussof, K. C. L. S. I. S. Convolutional neural network for person and car detection using yolo framework. *JTEC:http://journal.utem.edu.my/index.php/jtec/article/view/3599*.

Mandhare, P., Kharat, V., and Patil, C. (2018). Intelligent road traffic control system for traffic congestion a perspective. *International Journal of Computer Sciences and Engineering*, 6:908–915.

Menezes, R. S. T. d., Magalhaes, R. M., and Maia, H. (2019). Object recognition using convolutional neural networks. In *Artificial Neural Networks*. IntechOpen.

Organization, W. H. (2008). Global status report on road safety.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Wada, K. (2016). labelme: Image Polygonal Annotation with Python. `https://github.com/wkentaro/labelme`.

Zear, A., Singh, P., and Singh, Y. (2016). Intelligent transport system: A progressive review. *Indian Journal of Science and Technology*, 9.