Car Brand Recognition and Detection Based off Logo Present

CSE 498 - Computer Vision

Professor Aparna Bharati

Fall 2021

Final Project Report

By Alexander Spivey and Rishika Gunda

1 Problem Statement & Motivation

As of 2019, the number of registered vehicles on the road within the US was approximately 284.5 million [1]. This growth has been projected to reach 291 million by 2022. With the increasing forms of transportation and complex modern transportation networks, the need for intelligent transportation systems to monitor traffic action, record vehicle information, and report abnormal traffic events arises. The autonomous vehicle industry also relies heavily on its system's capabilities to detect vehicles and record the driver's surroundings in the case of an accident. As such, there need to be ways of detecting vehicle make, model, and year. Implementing the entire system in one would be extremely expensive and time-consuming, as well as the current processing system lacking the specifications to handle gigabytes of computation. With the time constraint in mind and capabilities in implementing an effective model and processing the data, the proposed system aims to classify car brands based on their vehicle logos, if visible and not obstructed or obscured.

Vehicle Logo Recognition (VLR) can be used for vehicle behavior analysis to assist vehicle identification, which is increasingly essential in highway management. Due to the countless number of drivers and vehicles on the road, it may be challenging to identify and track said vehicle during an accident. It is imperative that during such situations, the system will be able to identify vehicle makes. Vehicle identification plays a vital role in monitoring traffic and public security management. In general, license plate recognition (LPR) works better at recognizing an automobile. However, license plate information may not always be available if it is forged, covered, missing, or if the system has a hard time getting the appropriate vehicle information. Therefore, using VLR would allow for a more specific search in combination with LPR.

2 Previous Work

2.1 SIFT

Previously, some methods have been proposed for vehicle logo detection (VLD), but due to the numerous variances in texture, features, colors, and shapes of the logos, it was not easy to do so [2]. Wang et al.[2] used LPR as the base of the pipeline to detect the region where the logo resides, as it is common for the two to be in near proximity to one another. It is assumed that if a license plate is located accurately, then the rough location of the vehicle logo region could be determined. After the region has been proposed, morphological operations such as edge detection and color segmentation are applied to detect the logo [3, 4] accurately. However, this method is unreliable because the distance between a license plate and the vehicle logo is not constant nor a guarantee. To alleviate this problem,

many have turned to Scale Invariant Feature Transform (SIFT) based matching. Study [5] used the coordinates of the detected license plate to extract the vehicle mask. A phase consistency feature map is used to search the middle of the vehicle mask for the vehicle logo location [6]. Finally, multiple feature matching is used to recognize and detect the vehicle logo.

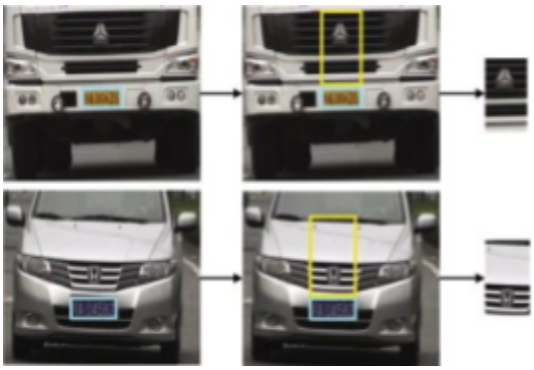

Fig. 2 (left): The process of vehicle logo coarse localization [7]

Fig. 3 (right): Correct detection of logo based on LPD method [7]

While this methodology fixes some of the proposed issues, it relies heavily on a symmetrical front view image of the vehicle and fails easily once the picture is tilted. It is also noted that if the vehicle mask is not detected or located correctly, the next step for logo recognition will fail. This failure could be due to poor illumination conditions, weather, dirt, partial occlusion, and different camera angle view/lens.

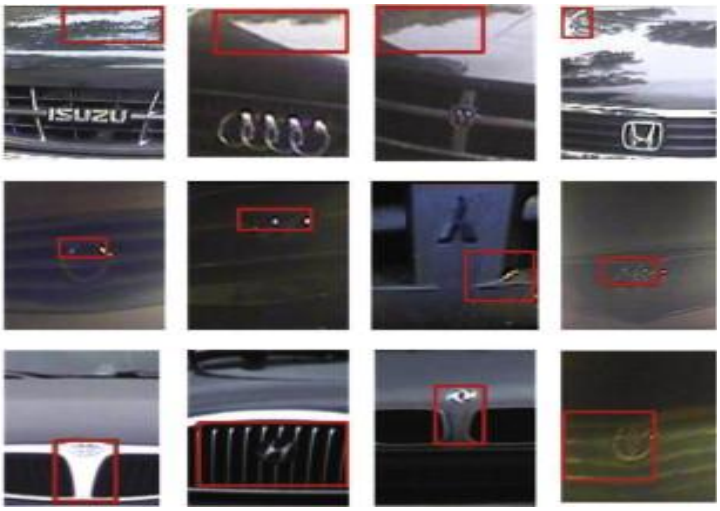

Fig. 4: The first, second and third rows represent the cases of false detection in sunshine, at twilight and nighttime, and the correct detection with unwanted background information [7]

Traditional VLR mainly included feature points, template matching, and edge features. Template matching was the most basic method used to classify different vehicle manufactures but was limited to only parallel changes in logo. Any alterations in terms of orientation or even zoomage would cause failure; such failures calls for the use of SIFT for orientation and scale issues, with the only drawback in the proposed algorithm being that the resolution of the input image must be fixed and edge features are unreliable when the pixel resolution of the image is low [3].

2.2 CNN

Previously, logo detection and classification was led primarily by keypoint-based detectors and descriptors. Researchers from the University of Milan Bicocca proposed a new pipeline model that consists of a "recall-oriented logo region proposal, followed by a Convolutional Neural Network." Different machine learning techniques were applied to the training set to increase recognition performance, "such as image pre-processing, class-balancing, sample weighting, and synthetic data augmentation" [8].

"Convolutional Neural Networks (CNN, or ConvNet) are a class of deep, feed-forward artificial neural networks that have successfully been applied to analyzing visual imagery" [9]. CNNs consist of input and output layers and many other hidden inner layers that handle data processing throughout the lifetime of a classification. These hidden layers entail convolutional layers, pooling layers, fully connected layers, and normalization layers. As shown in Figure 1, CNNs have three axes for how neurons are arranged within layers; being depth, width, and height. Neurons inside the various layers of the CNN are connected to the layer before it by several threads, which sit inside of a receptive field. Layers are also quite malleable, and frequently a CNN involves the conglomeration of a high number of layers that are interconnected in a variety of possibilities.

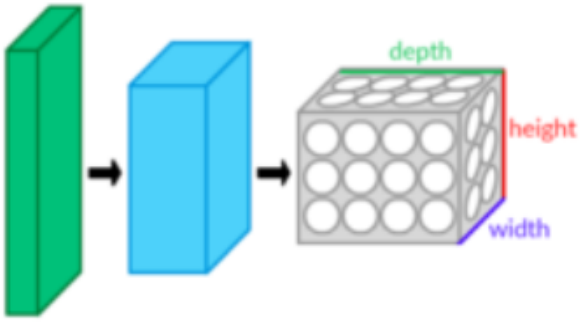

Fig. 5: Convolutional Neural Network [9]

As mentioned, CNN-based methods have gained extreme success in graphics and image processing tasks designed to accomplish the classification of different objects. A CNN-based system for VLR reduces the need to do precise logo detection and recognition. In [10], the proposed method consists of three main parts; The first step uses a constrained region detection model, Faster R-CNN, which is used to extract the car head and tail position details. The researchers used the VGG-16 network to obtain the feature maps with sufficient semantic information as the backbone for Faster-RCNN. The final feature maps are then used for the region proposal network. After that, the region of interest (ROI) pooling extracts the target area through the generated region proposals. The second part of the method is used for enhanced matching for small objects, consisting of constrained region segmentation and copy-pasting strategy. Vehicle logo regions can be cropped using corners of the detected candidate regions present in the constrained region and with the use of enhanced matching with a SSFPD(Single Shot Feature Pyramid Detector) network. They have used a reduced ResNeXt network for feature extraction, which uses a new feature pyramid with high-level semantic feature maps to avoid semantically weak features, which effectively improves recognition performance.

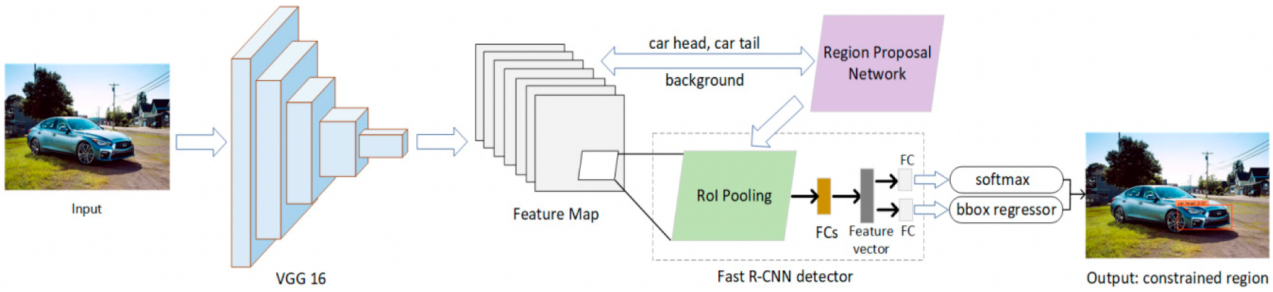

Fig. 6: Framework of the proposed network for vehicle logo recognition [10]

2.3 RCNN

In [11], Girshick used RCNN for object detection and then a linear classifier like SVM for object classification, which improved the mAP (mean average precision) for object detection by more than 30% compared to the best result on VOC 2012. They [11] combined a high-capacity CNN with bottom-up region proposals to localize and segment the objects present in an image. The working principle of the region proposal with CNN is as follows: As the name suggests, the first stage in RCNN is generating the region proposal or the regions in an image that could belong to a particular object. RCNN uses a selective search algorithm that works by creating a sub-segmentation of an

image. This segmentation can be based on color, texture, size, or shape. It iteratively combines similar regions in an image to form objects. This generates object proposals of different scales in an image. After generating various region proposals, CNN is used to generate feature vectors of 4096 dimensions for each region proposal in the image [12]. For each object proposal, mapping ground truth instances with maximum Intersection of Union (IOU) are labeled as positive, and the rest are treated as negative or background class. After that, the network samples randomly to ensure enough representation of positive and negative classes during the training. Once done, one final feature vector for each region proposal is generated, which is scored using object classification.

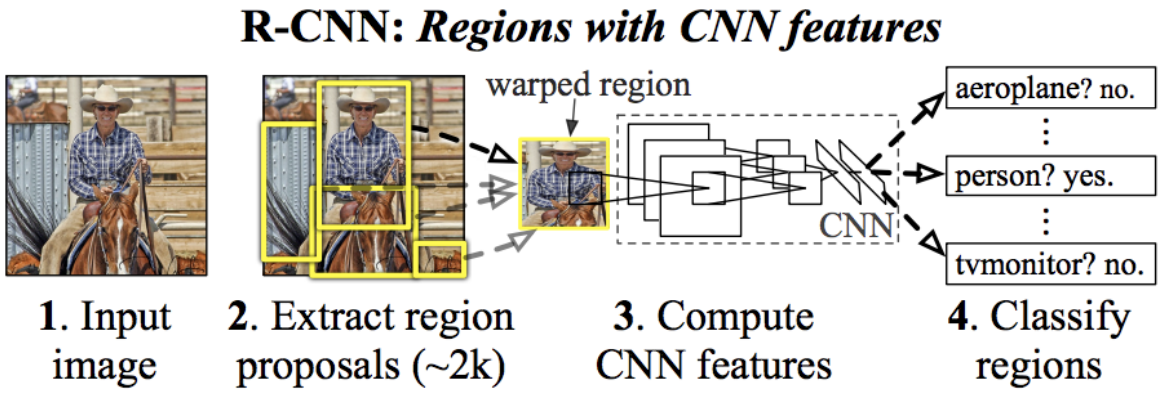

Fig. 7: RCNN architecture

2.4 Histogram of Gradients (HOG) with Linear Support Vector Machines (SVM)

HOG feature extraction is one of the many traditional Computer Vision techniques used for object recognition. In brief, gradient intensities of an image can lead to the recognition of an object in an image. HOG features are invariant to geometric and optical deformations in the image. HOG is implemented by grouping pixels into small cells, computing each gradient direction, and then converting them into orientation bins. This operation has to be done in both the x and y-directions. Each cell's gradient magnitude is then summed, with the biggest gradients having the most weight within their respective bins, while the smallest orientations are removed as noise. The resulting product is a histogram representing the dominant orientation of that particular grouping of pixels [13, 14]. These cells represent the structure of the image while allowing for variations. To better the model's performance by making our model more robust to changes in brightness and darkness, block normalization has to be applied by taking overlapping cells and combining their resulting HOG. Once this has been achieved, a normalization value is calculated and applied to normalize the resulting

histogram [13, 14]. Afterward, any feature classifier can be used, but most researchers favor SVM for its computational success.

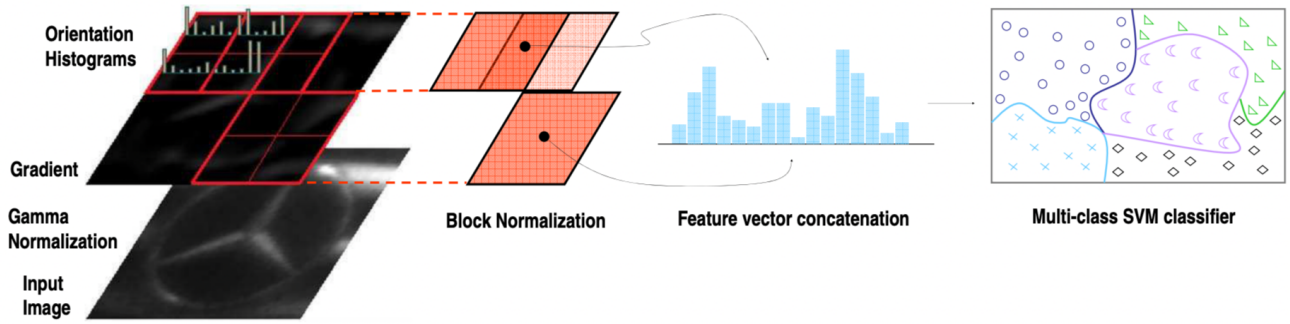

Fig. 8: Overview of the HOG/SVM architecture[15]

3 Dataset Used:

The team's initial plan was to use the clothing brands dataset to detect the logos and counterfeits of the logos, which led to the team's discovery of the LogoDet-3K dataset [16]. It contains nine overarching directories: clothes, electronics, food, leisure, medical, necessities, sports, and transportation. Within just the transportation folder, it contains another 213 different companies; each image has an associated annotated XML file that provides the object class and the bounding boxes of the logo. However, after much deliberate thought, the team realized that there are not many counterfeit logos within the clothing industry; rather, fake merchandise instead.

After various brainstorming sessions, the team decided to use car logos and full car images to detect and classify various car images based on their logos. From the LogoDet-3k dataset, the team found seven potential car brands for the first trial run: Bentley, Toyota, BMW, Lamborghini, Mercedes, Chevrolet, and Hyundai, but most of the logos present for each brand were blurry and distorted, so the team took it upon themselves to fix each annotation and replace/download additional images. Later on, Volkswagen was added to the training and testing sets.

After figuring out the brand logos, the team looked for a potential full-frontal car view dataset matching the seven selected brand cars. Most datasets like CompCars and DVM [17] lacked the brands chosen. The other dataset that matched the selected brand requirements was of low quality due to distortion and blur. After a couple of days of searching the web, the team decided that it would be best to manually create the dataset by scouring the web for high-quality images that matched our specifications. The resulting dataset contained a minimum of 50 images per class and had annotations that included all attributes of interest in the same format as the logos to prevent any parsing issues.

4 Approach

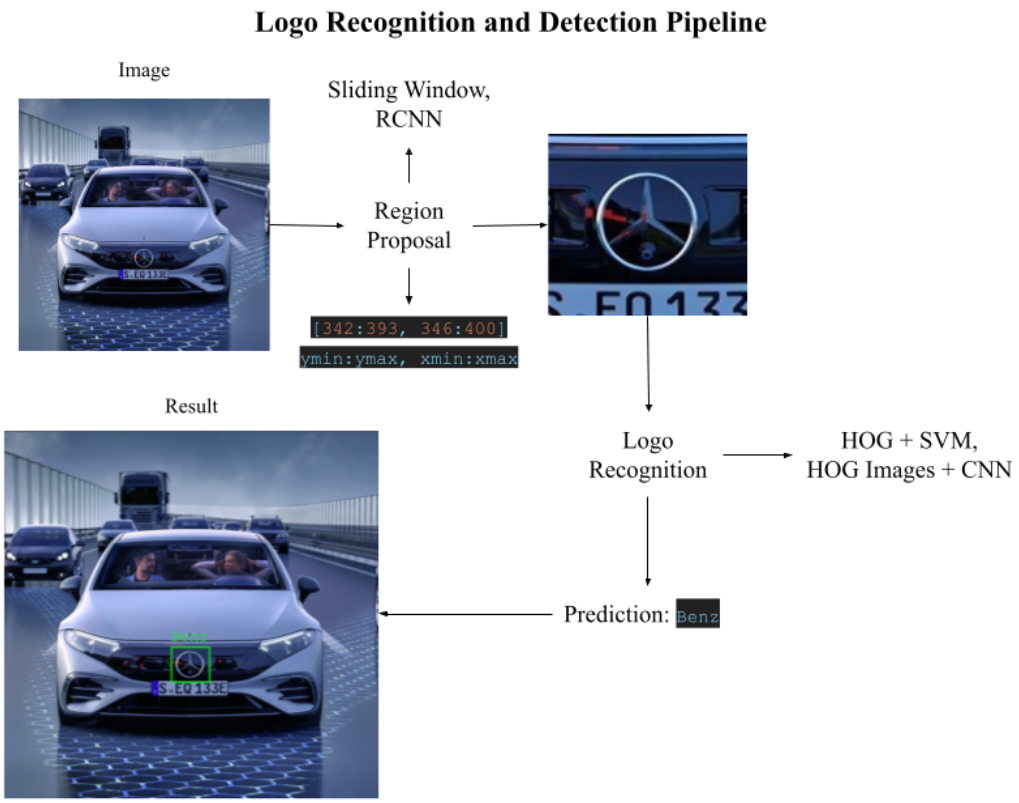

Fig. 9: Pipeline for the model used

It made sense to break each object's detection and recognition into two separate models to divide the problem on hand. For logo detection and recognition, the pipeline consists of an initial model to read the image in, and predict, where the object of interest's region may lie and another model to make predictions off the given region. The same pipeline/methodology is applicable for license plate detection and recognition. This paper will mention the issues encountered, but regardless, the code that has been written is more than applicable and can be used if modified and used with a proper dataset for logo training.

5 Experimentation and Exploration

The initial start of research involved finding previous models used, and how they varied in their strengths and weaknesses, this created the resulting Previous Work section above. After which, the team then focused on finding appropriate datasets to build the research off of. It was then that we found the aforementioned LogoDet-3k and chose the original logo. Due to the XML format being Pascal's VOC, it was imperative to create a parser that could read the XML and load the image

associated with the file and save each XML's object into a DataFrame so that the previously mentioned attributes could be referenced with ease. The parser and the models created to recognize the logo can be found in the Jupyter Notebook titled "Analysis of Directories - HOG + SGD or CNN w. Sliding Window." The resulting car brand's DataFrames created contained each image's: filename, file path, size, objectType, bndbox (bounding box), and bb_img (bounding box image). However, since most of this information is unnecessary during training, each DataFrame was then resaved with only the filename, bndbox, and objectType and then combined into one large DataFrame.

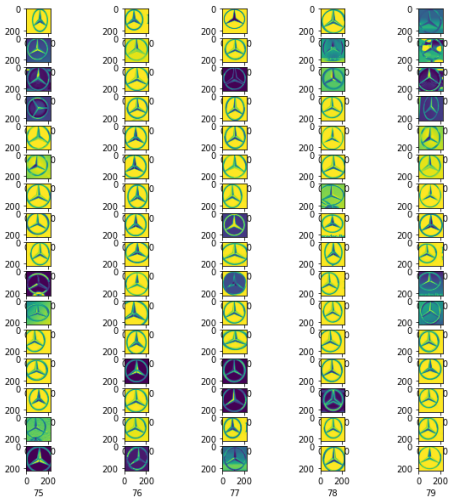

Fig. 10: Display of all Bounding Box Regions within the Benz class

One of the previous works mentioned earlier stated the use of HOG along with SVM. Due to the simplicity of the method, it was the first implementation used for logo recognition. The first step required extracting the HOG features and images from all the bounding box images. As such, a loop was used to append each image's resulting features and HOG images to a list for later use. It is important to mention at this point that all images were loaded in as grayscale due to color not being a recognition feature, and the objectType column was converted using a LabelEncoder, so each car's label would be numerical instead of categorical. The resulting HOG features were then used as the X and the associated labels as the Y for the train-test split. The training data was then fed into a basic implementation of SVC (Support Vector Classifier) and tested with different kernels. The resulting best kernel was linear, which allowed the team to use SVM's LinearSVC model and SGDclassifier. The first model scored consistently above 95%, with minor and understandable mistakes, and the latter model's best scored above 99%. However, when the latter was re-implemented, it never gave the same results due to the randomicity of SGD training.

Another favorable method for object recognition would be using a CNN; known for its outstanding accuracy and hunger for data, it lacked the simplicity to be diagnosed if the returned results were abysmal. The bounding box images of the logo were used as the training images and were then fed into a simple CNN. The resulting model never scored above 83% accuracy. The only options left to increase the model's effectiveness were to give it more data or to add more layers, with dropouts to prevent overfitting. Instead, curiosity took the best of one of the team members, Alexander Spivey, and tried feeding the network the HOG images instead of the grayscale logos used previously. The resulting model consistently scored above 97% accuracy even though no additional layers were added, no different epochs were used for training, and the same amount of training instances were used both times. After testing the effectiveness of HOG images and CNN some more, it became apparent that this model outperformed that of the HOG features and SVM and would outperform their counterpart CNN. The reason for the success of HOG images and CNN over that of the standard CNN model is believed to be that using HOG to simplify the logo image to its barebone structure allowed the CNN's random kernel to score better as it would not pick up on noise and had been given a limited amount of features to work with, in comparison to the logo's grayscale image. As of now, there are no other works that have mentioned this particular methodology.

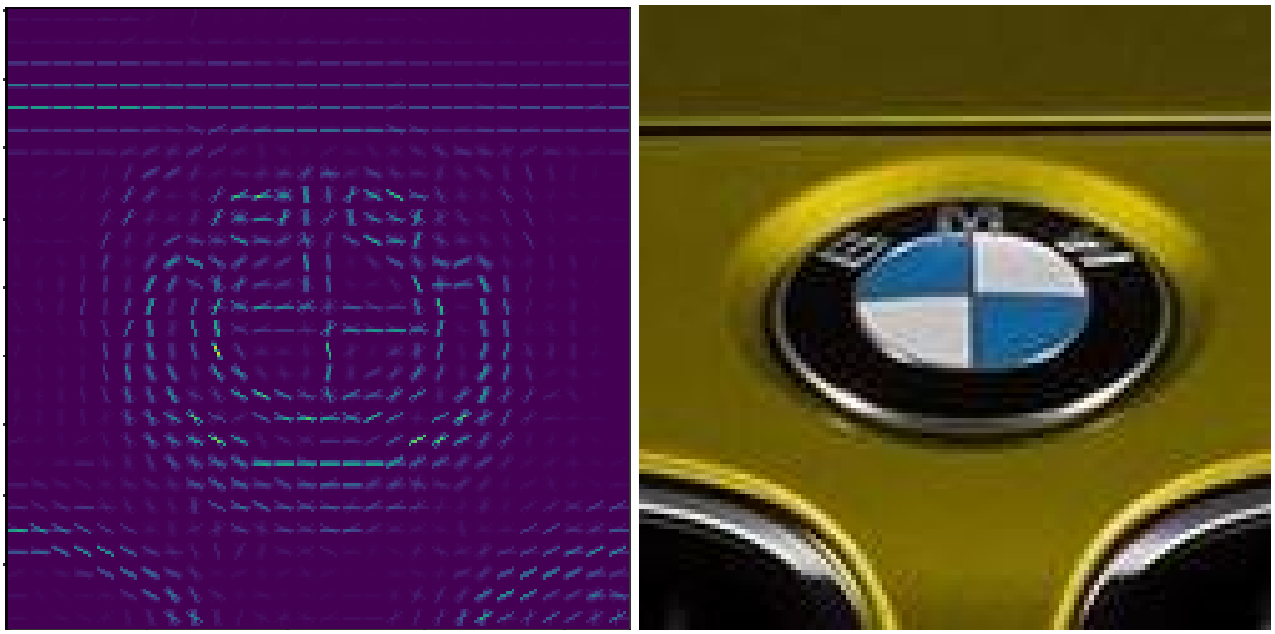

Fig. 11: HOG of Image (left) vs Original Image (right)

At this point, the team had a model capable of logo recognition with an accuracy of around 98% on average. The only other missing part in the pipeline was logo detection. Choosing simplicity over efficiency, the team decided on implementing a sliding window for detecting the region of interest. The sliding window would work by grabbing a window from the original image of the same input size used within the logo recognition model. This would then happen all the way across and down the image and at different image scales, such that if the logo were too large to be picked up on the original input size, it would eventually do so at a smaller resolution/scale. As mentioned, this model, while relatively simple, is extremely time-consuming and, to the team's dismay, not very effective. When used on any given image, the car's outline would always be detected and guessed as Bentley, which was later removed from use, and the actual logo's region was never predicted. The false prediction issue was countered by adding a background class to both logo recognition models, whereas the issue of predicting the logo's region was never fixed. Regardless of what scale or image was used, no correct prediction ever resulted. Due to the outlandish runtime needed per image and the lack of progression no matter the tuning, the sliding window methodology was discontinued for the logo region proposal.

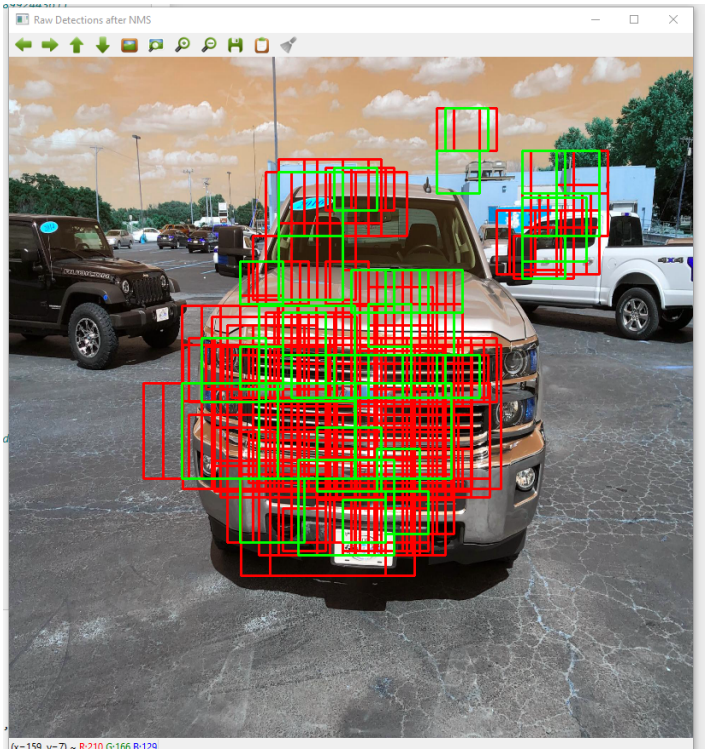

Fig. 12: Resulting regions of interest. Red regions are all predictions given and green regions are the chosen non max suppression results

The other work mentioned for the region proposal was the use of RCNN. The team first bit off more than it could chew by choosing to implement Faster RCNN initially. The team eventually ran into coding difficulties and eventually abandoned the implementation due to the limited time frame. Regardless, Faster RCNN can detect ROI within five frames at times, which is more than over qualified for the task on hand. RCNN works by using OpenCV's selective search to get a multitude of regions for proposal. Below is the output of such an example.

Fig. 13: The left image is the base image and the right contains all results (green boxes) given by selective search

To train an RCNN, all car images were loaded into their respective DataFrames, 50 images per class. Each image was then selected for selective search. Once all results were compiled for an image, each resulting region was compared with the ground truth bounding boxes of the logo present on the car. If the IoU(Intersection of Union) of the bounding box and proposed region score above 70%, its image then gets appended to the training list, with its respective index in the label list being '1'. Any regions that hit this score are considered 'true logos.' If the score instead was below 30%, it was also appended to the training list but labeled '0' instead. Any regions that hit this score are considered 'false logos.' The original first testing model contained all six previously mentioned car brands, except Bentley, which was later replaced with Volkswagen and had a 1:30 ratio for true to false logos.

Fig. 14: The images above are actual results of true (left) and false (right) logo regions created from the proposed method

The RCNN model was built off the backbone of VGG16 with the weights of ImageNet. Due to the team's limited computer specifications, the batch size was limited to a maximum of 10 images, with steps per epoch of 20. Testing was applied to the original car images by loading each image as the base image and conducting a selective search to get all-region proposals. Typically the RCNN model limits region predictions to only 2000. However, after some testing, the cap was removed, as sometimes the accurate logo's proposal region did not constantly occur in the first 2000. However, by removing the limit of regions checked, each image testing time often took around 3-4 minutes per image but received a higher accuracy. Any regions that scored confidently above 70% were appended to a list of all ROIs, along with their respective confidence level. In the end, the max confidence prediction was chosen as the final proposed region, and then any of the two recognition models can be used on the region to predict the logo present. The team's preferred classifier was consistently the CNN trained on HOG images. To the team's delight, the initial model started predicting accurately for brands like Benz with ease, but never accurately for BMW or Lamborghini. This was due to those brands' logos being on the hood of the car instead and being extremely small compared to other brands. As a result, In the end, the team decided to make three different pipelines, with variations ranging in the number of class brands trained on to the ratio of actual to false logos given to the RCNN.

Fig. 15: Comparing the size, placement, and orientation of BMW's logo (left) against Benz (right)

6 Results

Pipeline Results

Pipeline Number & Number of Images Tested	Classes Trained On	Ratio of True Logo to False	RCNN Model Name	CNN Model Name	Correct Predictions / Predictions Made
Pipeline One 105 images	[Benz, Chevrolet, Toyota, Volkswagen]	1:5	‘model5moreImages’	‘cnnHOGwRegion3’	61/96
Pipeline Two 105 images	[Benz, Chevrolet, Toyota, Volkswagen]	3:4	‘model4moreImages’	‘cnnHOGwRegion3’	42/105
Pipeline Three 177 images	[Benz, BMW, Chevrolet, Hyundai, Lamborghini, Toyota, Volkswagen]	1:5	‘modelALL’	‘ALLcnnHOGwRegion’	41/172

# of True Predictions vs False Predictions (False Positives included as TRUE)

Pipeline #	Benz	BMW	Chevrolet	Hyundai	Lamborghini	Toyota	Volkswagen
One	14:12	N.A	16:8	N.A	N.A	17:5	21:3
Two	8:19	N.A	7:19	N.A	N.A	10:14	21:6
Three	13:14	9:15	3:23	9:16	1:23	3:22	5:22

# of True Predictions vs False Predictions

Pipeline #	Benz	BMW	Chevrolet	Hyundai	Lamborghini	Toyota	Volkswagen
One	14:12	N.A	15:11	N.A	N.A	14:7	15:9
Two	8:19	N.A	6:20	N.A	N.A	8:16	12:15
Three	12:16	0:26	0:26	3:25	0:24	1:24	3:24

The chart above explains each pipeline's specifications and the associated models' names, which are saved and can be quickly loaded. It is easy to see that pipeline one had the best results overall and pipeline three had the worst performance, with most correct predictions being false positives, as the predicted region was incorrect, while the logo prediction was still correct. Based on the pipeline results, the assumption that BMW and Lamborghini cannot be predicted due to the odd placement and smaller size was correct acted as additional noise within pipeline three, causing region predictions to vary drastically regardless of the image tested on. This, however, does not explain why pipeline three does not accurately predict Hyundai regions. Looking back at the dataset accumulated for Hyundai, the images drastically differ in view. As such, the RCNN model did not have enough data to learn off of to predict pictures that are off just the car front, which was the testing images used. The pipeline results increased significantly by removing these three classes when maintaining the same 1:5 logo ratio. When the ratio was made more even, 3:4, pipeline 2 did significantly worse, with most correct predictions being false positives. Based on this result, it is assumed that RCNN models need to know more false cases than true, which makes sense in terms of image area to actual object area; the logo region only appearing once on every image and being only a small portion of the image. After testing on these three pipelines, it becomes apparent that the RCNN prefers logos that are large and centered on the car's grill, have fewer classes to help prevent false positive calls on regions of non interest, and prefer an uneven distribution of true to false logo regions to be trained on. While the team wanted to implement more variations on the model to test results, each training and testing on a pipeline took approximately 8-16 hours.

7 Concluding Thoughts and Future Work

The model could have been better trained if more images had been gathered and a better computer had been used for pipeline development. Due to RCNN being the gateway for CNN to make predictions, this pipeline heavily relied on RCNN's capability to predict accurately. Any further development should be focused on increasing the region proposal accuracy and testing the effect of training CNN with HOG images instead, as this was a completely new discovery during experimentation. Throughout the project assignment, the team ran into multiple issues such as changing project proposals, searching for a usable dataset, and having a team member drop out. While these unpredictable factors impaired the overall progress and success of the project, the team was still able to reach their original goal of creating a logo recognition and detection pipeline, as long as prediction time is not constrained.

8 References

1. *US VIO vehicle registration statistics: See how many cars in the US*. Hedges & Company. (2021, November 4). Retrieved November 29, 2021, from https://hedgescompany.com/automotive-market-research-statistics/auto-mailing-lists-and-marketing/.
2. Y. Wang, Z. Liu and F. Xiao, "A fast coarse-to-fine vehicle logo detection and recognition method," 2007 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2007, pp. 691-696, doi: 10.1109/ROBIO.2007.4522246.
3. W. Li and L. Li, "A Novel Approach for Vehicle-logo Location Based on Edge Detection and Morphological Filter," 2009 Second International Symposium on Electronic Commerce and Security, 2009, pp. 343-345, doi: 10.1109/ISECS.2009.251.
4. Y. Liu and S. Li, "A vehicle-logo location approach based on edge detection and projection," Proceedings of 2011 IEEE International Conference on Vehicular Electronics and Safety, 2011, pp. 165-168, doi: 10.1109/ICVES.2011.5983808.
5. A. P. Psyllos, C. E. Anagnostopoulos and E. Kayafas, "Vehicle Logo Recognition Using a SIFT-Based Enhanced Matching Scheme," in IEEE Transactions on Intelligent Transportation Systems, vol. 11, no. 2, pp. 322-328, June 2010, doi: 10.1109/TITS.2010.2042714.
6. P. D. Kovesi, “Image features from phase congruency,” in Videre: A Journal of Computer Vision Research. Cambridge, MA: MIT Press, 1999, pp. 1–27.
7. Mao, S., Ye, M., Li, X., Pang, F., & Zhou, J. (2013, April 6). Rapid vehicle logo region detection based on information theory. Computers & Electrical Engineering. Retrieved November 17, 2021, from https://www.sciencedirect.com/science/article/pii/S004579061300061X.
8. Bianco, S., Buzzelli, M., Mazzini, D., & Schettini, R. (2017, March). *Deep learning for logo recognition*. Science Direct. Retrieved November 29, 2021, from https://doi.org/10.1016/j.neucom.2017.03.051.
9. Chandana N, Harshitha M, Chondamma P S, Anusha M, Dr. S. Padmashree, 2018, Brand Logo Detection Using Convolutional Neural Network, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NCESC – 2018 (Volume 6 – Issue 13),

10. Liu, R.; Han, Q.; Min, W.; Zhou, L.; Xu, J. Vehicle Logo Recognition Based on Enhanced Matching for Small Objects, Constrained Region and SSFPD Network. Sensors 2019, 19, 4528. https://doi.org/10.3390/s19204528
11. R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 2014 pp. 580-587. doi: 10.1109/CVPR.2014.81
12. Gandhi, R. (2018, July 9). R-CNN, fast R-CNN, Faster R-CNN, YOLO - object detection algorithms. Medium. Retrieved November 25, 2021, from https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e.
13. D. F. Llorca. (2021). Vehicle logo recognition in traffic images using HOG features and SVM. Vehicle Logo recognition. Retrieved November 17, 2021, from http://www.robesafe.com/personal/roberto.arroyo/docs/Llorca13itsc.pdf.
14. Burkhard, T. (n.d.). Vehicle Logo Recognition and Classification: Feature Descriptors vs. Shape Descriptors. Vehicle Logo recognition. Retrieved November 17, 2021, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.208.2068&rep=rep1&type=pdf.
15. Ananth, S. (2020, September 30). R-CNN for object detection. Medium. Retrieved November 25, 2021, from https://towardsdatascience.com/r-cnn-for-object-detection-a-technical-summary-9e7bfa8a557c.
16. Jing, W. (2020, September 22). *Logodet-3K*. Kaggle. Retrieved November 29, 2021, from https://www.kaggle.com/lyly99/logodet3k.
17. Jingming Huang, Bowei Chen, Lan Luo, Shigang Yue, Iadh Ounis (2021). "DVM-CAR: A large-scale automotive dataset for visual marketing research and applications". In: ArXiv e-prints (Aug.2021). arXiv: 2109.00881