

APS360 Final Report
Date: August 9, 2020

Group 33: The Pursuit of Accuracy, The Final Showdown

Group members:

Andy Jiang (1003886039)
Deep Pandya (1003140585)
Sophie Kim (1003198773)
Johnathon Martin (1004350230)

Word Count: 2287

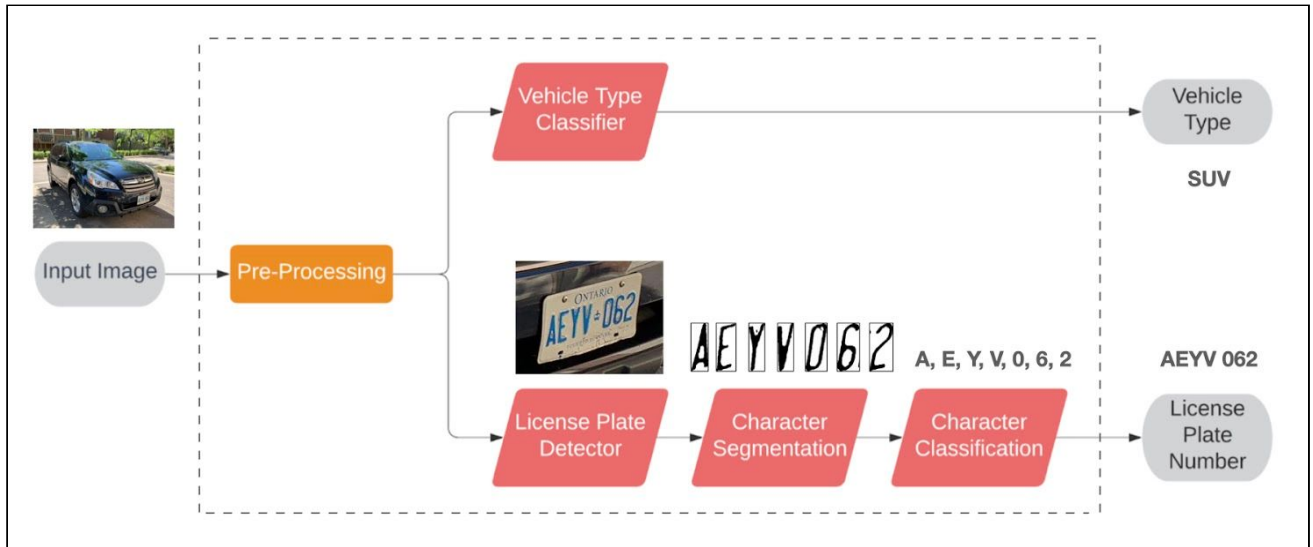
Introduction	2
Illustration	2
Background and Related Work	2
Data Processing	3
License Plate Locator	3
Character Recognition	3
Vehicle Type Classification	3
Architecture	5
License Plate Locator	5
Character Recognition	5
Vehicle Type Classification	6
Baseline Model	6
Character Recognition	6
Vehicle Type Classification	7
Quantitative Results	7
Character Recognition	7
Vehicle Type Classification	8
Qualitative Results	9
License Plate Locator (Faster R-CNN)	9
Character Recognition	10
Vehicle Type Classification	10
Evaluate Model on New Data	11
Character Recognition Model Evaluation	11
Vehicle Type Classification Model Evaluation	11
Full System Testing	12
Discussion	15
Ethical Considerations	15
Project Difficulty	15
References	16
Part B: Individual Contribution	17
Overall Contribution Scores	17
Individual Contribution Summaries	17

1. Introduction

The goal of this project was to create a system that can accurately identify defining characteristics of cars, such as the license plate number and vehicle type. This type of multifaceted identification system is applicable to many real-world scenarios such as improving efficiency and accuracy of police search and automation in parking lots. Machine learning is perfect for this project because of its ability to learn and decipher images with variations in angles, location of a car within an image, and lighting conditions.

2. Illustration

The system follows two separate data flow paths: the vehicle type classifier path and the license plate identifier path, which consists of the license plate detector, character segmentation, and character classification. The input images are pre-processed using computer vision and PyTorch transforms.



3. Background and Related Work

Automatic Number Plate Recognition generally involves four steps: pre-processing, license plate detection, character segmentation, and character classification [1].

Kaur et al and Xie et al use similar pre-processing and license plate detection methods. Given an input image of a vehicle, their networks output the cropped license plates with 99.33% [1] and 95% [2] accuracy respectively. Numerous image pre-processing techniques, such as colour correction, noise reduction, contrast enhancement, and edge detection are utilized to isolate the plate area.

Xie et al delve into character segmentation and classification as well. For accurate character isolation, using a binary image, horizontal and vertical projection integral values are used to plot histograms that indicate regions of high character density. For character segmentation, they count the number of changes between white and black pixels in each column of pixels in the binary image. Using these values, a histogram is plotted to show distinct regions, indicating a character on the plate. For character recognition, density features and edge distance features are inputted into a backpropagation neural network. The vertical and horizontal density features represent the width and height of the character respectively. This model yields a 97.7% accuracy. [2]

For vehicle type classification, detection of useful features without noise is difficult, which is why in Hao Lyu's study, a four-layered CNN is used. It takes a cropped side view of the vehicle as input and yields an accuracy of 97%. [3]

4. Data Processing

4.1. License Plate Locator

For the license plate locator, we used the OpenALPR (Automated License Plate Recognition Dataset) [4] consisting of 222 US license plate images with a corresponding text file containing bounding box locations. We created a CSV file, aggregating bounding box, class, and image path information for all images in the dataset and used this as input to train the Faster R-CNN.

Example of data in CSV:




filename	width	height	class	xmin	ymin	xmax	ymax
wts-lg-000121.jpg	76	38	plate	880	92	956	130
wts-lg-000140.jpg	85	42	plate	716	205	801	247

4.2. Character Recognition

For the character recognition model, we used the Chars74k dataset [5], which contained images split evenly into 62 classes (0-9, a-z, A-Z). Lowercase letters were out of scope and thus removed, leaving 36,576 images consisting of 36 classes. This dataset was then split into training, validation, and testing sets:

Number of training images (70%)	Number of validation images (15%)	Number of testing images (15%)
25,596	5,508	5,472

Every character passed into our model was normalized and grayscaled, as the image colours are not relevant to the character it represents. We applied random rotations and perspective changes on the training data to better emulate the problem of reading license plates at different angles.

Before (RGB)	GreyScale	Rotation & Perspective Change
		

4.3. Vehicle Type Classification

The vehicle type classification model was trained using the Stanford Cars Dataset [6], which consists of 16185 car images and 196 classes. In this dataset, the images are classified by their vehicle make, model, and year. It also contains a .mat datasheet that maps the index of the corresponding class to the class name and another .mat dataset that consists of relative image file paths, bounding box coordinates for each image, index of the corresponding class, and whether or not the image is in the original test set.

To format the data, the 196 original classes were simplified or, in some cases, reclassified as one of our seven defined vehicle types: hatchback, sedan, SUV, truck, van, convertible, and coupe. When reclassifying, we used a voting system, where the majority vote won.



Original Classification	Simplified Original Classification	Finalized Classification	Votes			
			Deep	Andy	Johnathon	Sophie
'AM General Hummer SUV 2000'	SUV	SUV	Simplified Original Classification is in one of our 7 classes			
'Acura RL Sedan 2012'	Sedan	Sedan				
'Acura TL Sedan 2012'	Sedan	Sedan				
'Acura TL Type-S 2008'	S	Sedan	Sedan	Sedan	Sedan	Sedan
'Acura TSX Sedan 2012'	Sedan	Sedan	Simplified Original Classification is in one of our 7 classes			
'Acura Integra Type R 2001'	R	Coupe	Coupe	Coupe	Coupe	Sedan
'Acura ZDX Hatchback 2012'	Hatchback	Hatchback	Simplified Original Classification is in one of our 7 classes			

Each image of the car was also cropped using OpenCV and the provided bounding boxes.

The number of pictures of each car type after reclassification is as below:

Hatchback	Sedan	SUV	Truck	Van	Convertible	Coupe
1,606	4,122	2,935	1,514	1,149	2,058	2,801

An example of a cleaned class/label associated with a processed image:

	Before	After
Image		
Label	Acura TL Type-S 2008	Sedan

To balance the dataset, the images in each class folder were augmented until there were 5000 images per class, for a total of 35000 images. Augmentations included: horizontal flips, rotations between -8 to 10 degrees, and horizontal and vertical translations. After balancing, the dataset was split into train, validation, and test sets:

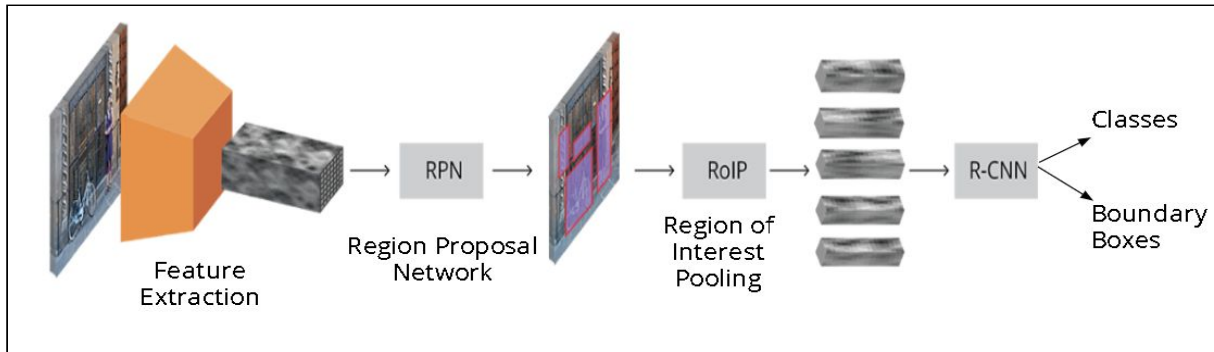
Number of training images (60%)	Number of validation images (20%)	Number of testing images (20%)
21,000	7,000	7,000

New testing data collected for model evaluation is discussed in section 9.

5. Architecture

5.1. License Plate Locator

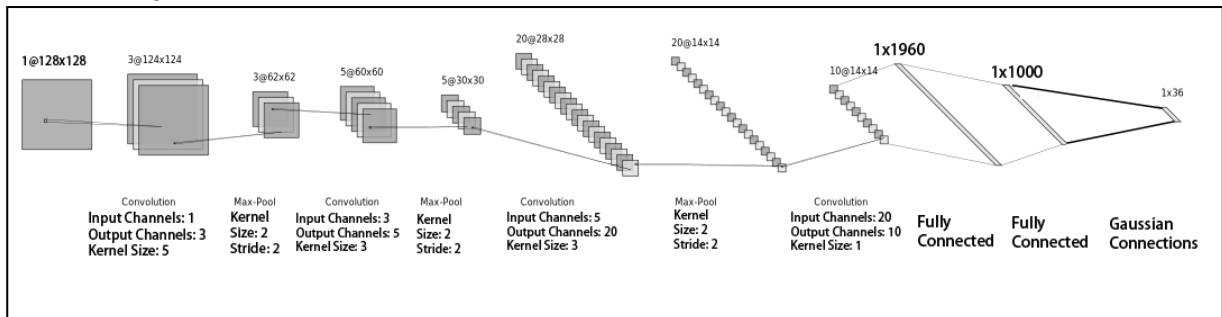
Trained a Faster R-CNN using the pre-trained ResNet50 FPN to locate plates within an image.



Source: [7]

5.2. Character Recognition

For character recognition, our team implemented a custom CNN with the architecture with the following architecture:

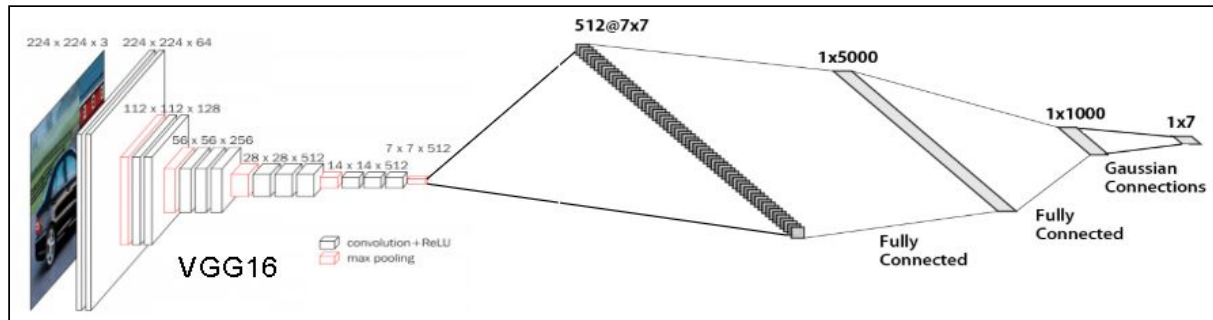


Below is a summary of the hyperparameters used to yield the best results for this model:

Loss Function	Cross-Entropy Loss
Optimizer	Adam
Activation Function	ReLu
Learning Rate	0.001
Batch Size	16
Number of Epochs	12

5.3. Vehicle Type Classification

For classifying vehicle type, our team applied transfer learning. For our final model, we used a pre-trained VGG16 model for feature extraction. We then trained two fully connected layers to apply the pre-trained model to our vehicle type detection problem.



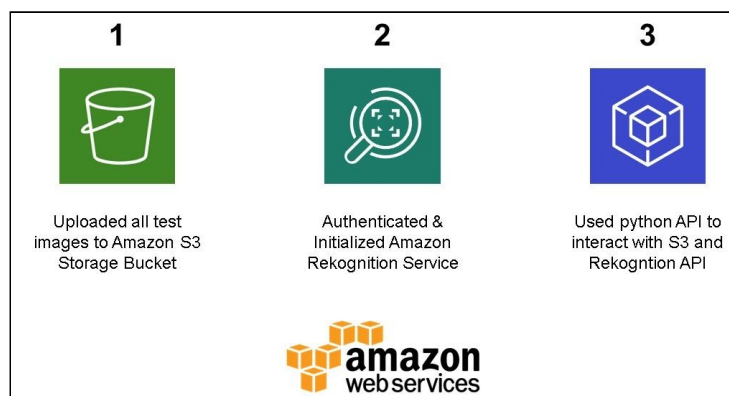
Below is a summary of the hyperparameters used to yield the best results for this model:

Loss Function	Cross-Entropy Loss
Optimizer	Stochastic Gradient Descent
Activation Function	ReLU
Learning Rate	0.007
Batch Size	20
Number of Epochs	30

6. Baseline Model

6.1. Character Recognition

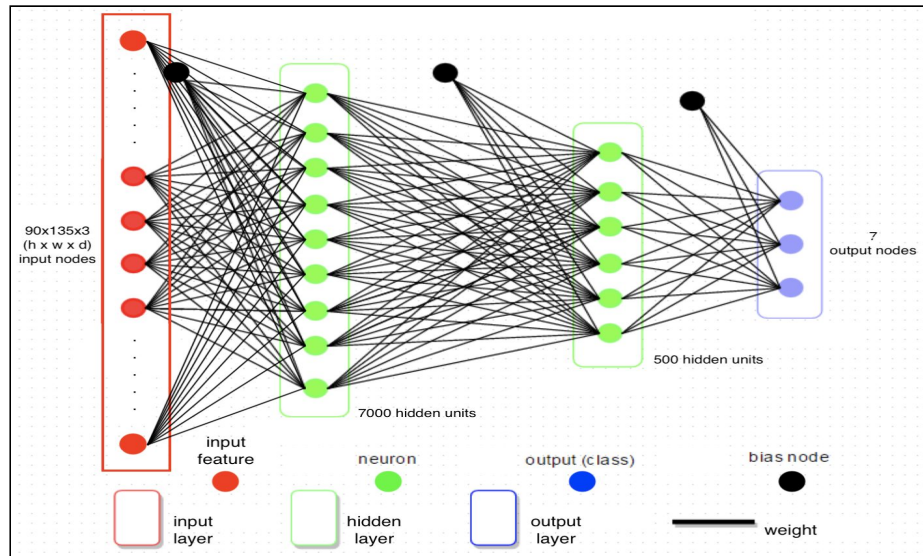
Our team used Amazon's Rekognition service as a baseline for the character recognition component of our project. Rekognition can detect text from images. This is a quick and easy baseline model as we just make API calls to the Rekognition service with the images we want to evaluate and it returns a list of detected text and their respective confidence levels.



Icon Source: [8]

6.2. Vehicle Type Classification

For the car type identifier baseline, we used a 3 layered ANN with a batch size of 700, a learning rate of 0.007, and 30 epochs due to its simplicity. 3 layers were used to mitigate the limitation of a single layer ANN when solving non-linearly separable problems.



Original Image: [9]

7. Quantitative Results

The quantitative measures we used to evaluate model performance were:

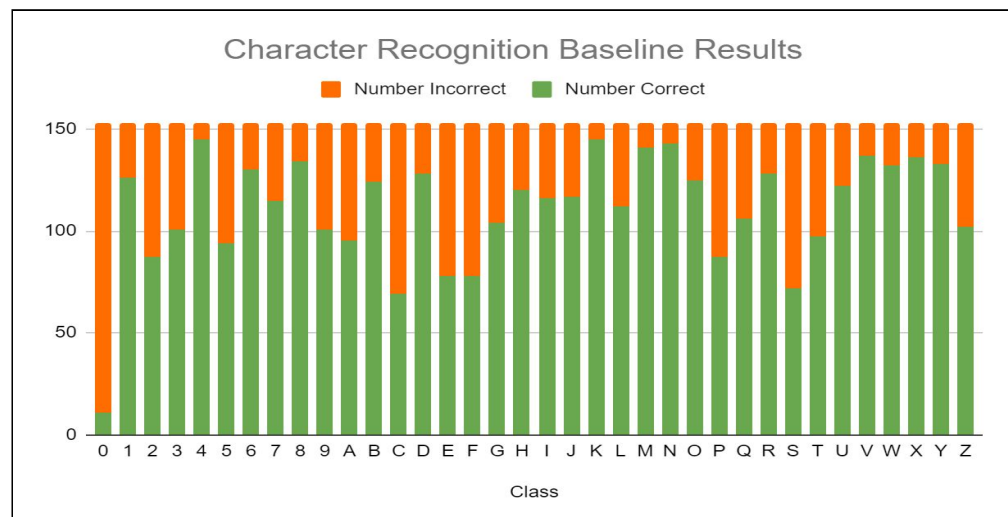
- Baseline model accuracy
- Model accuracy on the validation set
- Model accuracy on the testing set

where accuracy is percentage defined by the number of correctly classified images vs the total number of images.

7.1. Character Recognition

Baseline Model Results

Baseline Model Accuracy: 72.46%

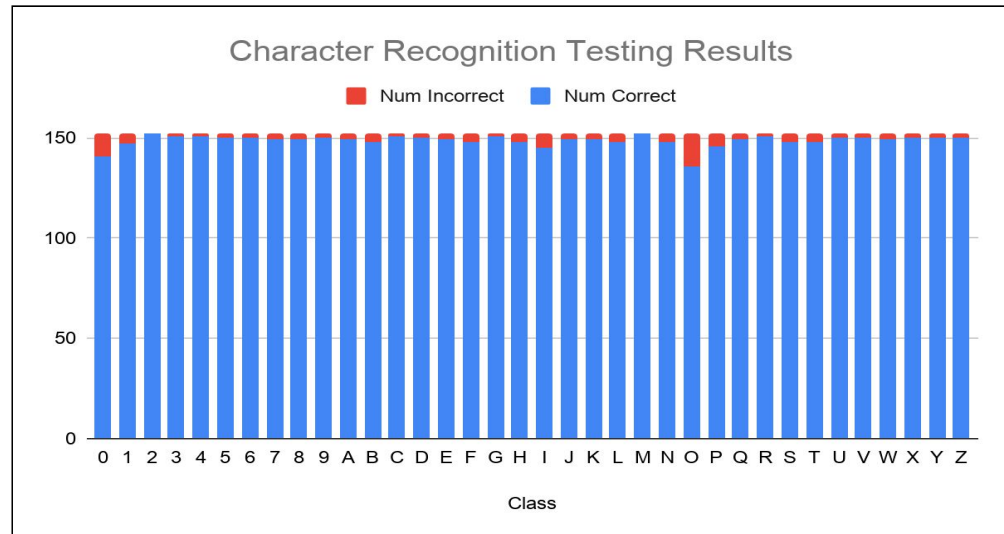


Training & Validation Results

Best Validation Accuracy: 98.1%

Testing Results

Testing Accuracy: 97.8%



The model performed very well for the majority of characters. The model struggled slightly with differentiating between '0' and 'O' but overall it was well above the baseline accuracy.

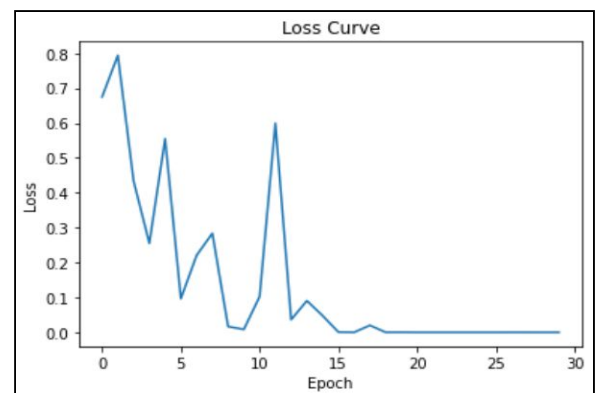
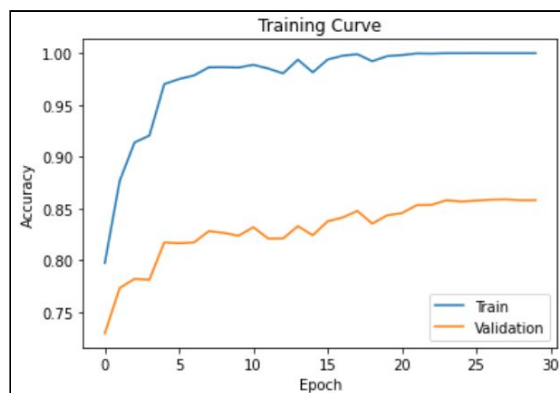
7.2. Vehicle Type Classification

Baseline Model Results

Baseline Model Accuracy: 44.81%

Training & Validation Results

Best Validation Accuracy: 85.89%



Testing Results

Testing Accuracy: 85.64%

Confusion Matrix

Our model did well in classifying trucks, vans, convertibles & SUVs as they all have distinct shapes that differentiate them from one another. However, our model could not differentiate between sedans and coupes as well, as they only differ by the number of doors.

		Predicted						
		Convertible	Coupe	Hatchback	SUV	Sedan	Truck	Van
Actual	Convertible	864	70	12	8	39	4	3
	Coupe	82	758	38	4	117	0	1
	Hatchback	9	45	806	42	83	1	14
	SUV	2	4	45	891	20	22	16
	Sedan	19	118	59	36	760	1	7
	Truck	3	0	2	43	3	947	2
	Van	0	0	9	10	8	4	969

8. Qualitative Results

8.1. License Plate Locator (Faster R-CNN)

Blue Bounding boxes were produced by trained faster RCNN.

Observations:

- Works well in different lighting conditions, angles and image sizes







Limitations:

- Does not work well on blurry or low-quality images

8.2. Character Recognition



Below are 4 images of correctly and incorrectly predicted characters from our model.

Prediction Inconsistent with Label	Prediction Consistent with Label
 <p>Predicted: F, Actual: P</p>	 <p>Predicted: P, Actual: P</p>
 <p>Predicted: 0, Actual: O This mix up is the most common error</p>	 <p>Predicted: O, Actual: O</p>

Our model does well predicting characters that hold a more traditional shape. Lots of characters from the dataset deviate from this shape to increase robustness. An example of this can be seen above with the 'P' wrongly predicted as 'F'. The other major limitation with the model is the inability to differentiate between 'O' and '0', this is due to the heavy overlap in the features shared between them.

8.3. Vehicle Type Classification

Below, there are 2 images of correctly and incorrectly predicted car types from our model. We can see that our model incorrectly predicted the left image as a coupe instead of a sedan. As discussed in the confusion matrix of the car type quantitative results, our model had some difficulty correctly distinguishing between coupes and sedans due to the similarity in the shape of the two types of vehicles. The table below visually shows this point in our confusion matrix. The only major distinguishing characteristic is the difference in the number of doors.

Prediction Inconsistent with Label	Prediction Consistent with Label
 <p>Model Prediction: coupe Actual Label: sedan</p>	 <p>Model Prediction: van Actual Label: van</p>

9. Evaluate Model on New Data

9.1. Character Recognition Model Evaluation

Our team used 15% of the original Chars74K dataset as a hold out set. This set contains 5,472 images that have not been seen by our model or used to tune hyperparameters in any way. Our model achieved an accuracy of 97.8% on this test set.

9.2. Vehicle Type Classification Model Evaluation

Data Collection

To evaluate our vehicle type classifier on new data, we chose to collect images of cars from Google Street View as this accurately reflects real-life data. We collected a total of 49 images (7 images per class). Below are a few images that we collected:



Results

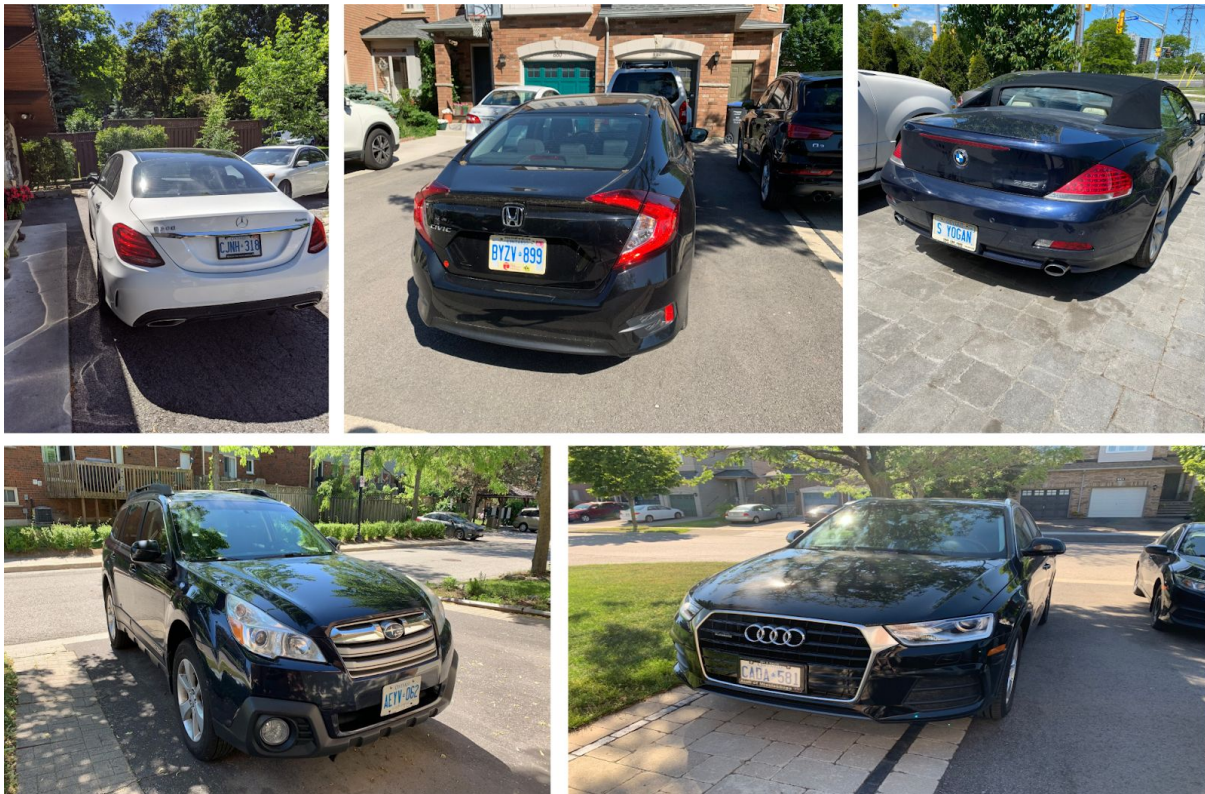
Our trained model achieved an accuracy of 81.6% on these images. This result is in line with the validation and testing accuracies that were produced by our model. This also tells us that our model is good at generalizing to new, never before seen data. From the confusion matrix, we can see that the model did not perform as well with coupes and hatchbacks as many of them were misclassified as sedans. This result is on par with what we saw in section 7.2.

		Predicted						
		Convertible	Coupe	Hatchback	SUV	Sedan	Truck	Van
Actual	Convertible	7	0	0	0	0	0	0
	Coupe	0	3	1	0	3	0	0
	Hatchback	0	0	4	0	3	0	0
	SUV	0	0	0	7	0	0	0
	Sedan	0	1	0	0	6	0	0
	Truck	0	0	0	1	0	6	0
	Van	0	0	0	0	0	0	7

9.3. Full System Testing

Data Collection

To test our full end to end system, we asked friends to send pictures of their vehicles from different angles with a visible license plate. Collected a total of 40 test images.



License Plate Locator Result

- Accuracy: 100%
- The system was able to correctly locate all license plates in the images

Character Segmentation







- Our system correctly segmented all license plate characters for 21 of the 40 images
- Breakdown of results:

Number of Successfully Cropped Characters	Number of plates
7	21
6	2
5	2
4	0
3	0
2	2
1	1
0	12

Examples of images where our segmentation algorithm picked up all 7 characters:



Examples of images where our segmentation algorithm failed (< 3 characters segmented):

		
		
Reflection on Plate (Causes most letters to be covered)	Shadow on Plate (Causes letters to be partially covered)	Dirty Plate Cover (Causes letters to be fully covered)

Character Recognition

In order to fairly evaluate the character recognition model, we only tested on the 21 images where our character segmentation was successful (segmenting exactly 7 or more characters).

Number of Characters Off	Number of Plates
0 (Perfect Match)	5
1	10
2	1
3	2
4	3

Our system correctly classified 79.59% of the characters from the 21 license plates that were tested. Although our system only perfectly matched 5 license plates, most of the time it was only off by one character. Our system can tolerate minor one or two character errors as we also provide vehicle type information. This combination allows for an effective vehicle identifier for police searches.

Vehicle Type Classification

Our system correctly classified 18/40 images. Our model performed poorly because of the quality of the images we collected. These two factors led to poor results:

1. Our model was trained on images that show the side profile of cars. However, most of the images that we collected were front or rear views with the license plate being in the picture.

2. Some of the images were taken in landscape and some were taken in portrait. When the portrait images are resized, they are stretched and warped. The portrait images were the cause of most of the errors.

10. Discussion

Overall, our team is happy with the way our project turned out. As discussed in sections 7 and 8, both our models were able to outperform our baseline models and performed well individually. We were able to see these levels of performance for the vehicle type classification model with the help of transfer learning. To ensure that the model is able to generalize, Section 9 discussed how the model performed on new, never before seen data.

In terms of the license plate locator model, we were delighted at how well our model worked despite only being given 222 images for training. We were surprised to see our Faster R-CNN model locating dark license plates, despite only being trained on plates with light backgrounds. This was an unexpected result.

Our character segmentation algorithm was the bottleneck in our system, as it used manual computer vision algorithms. As a future improvement, we could train a mask R-CNN or faster RCNN to locate characters within an image, which should be able to address the pitfalls experienced by the computer vision algorithms.

We learned that machine learning projects require a lot of experimentation and sometimes unexpected changes can yield good results. In addition to this, we also learned how to complete a project with finite compute resources and a finite amount of time.

11. Ethical Considerations

The ethical considerations arise during data collection and with model accuracy. When collecting data and taking pictures of personal vehicles, we recognize that sensitive information is captured, like the Vehicle Identification Number (VIN) and license plate number. To avoid ethical concerns, we will not be taking pictures of VINs, and license plates will only be used for development and deleted upon project completion. In real-world applications of our system, as we intend for our model to be used for police investigations, any errors could result in the pursuit of a wrong vehicle. To ensure the most accurate vehicle identification, we wanted to provide as much information about the vehicle as possible, which is why we decided on vehicle type classification in conjunction with license plate number identification.

The limitations of our model include poor performance on non-ideal images, such as low light, blurry, or obstructed images. Furthermore, the scope of our project will only support English license plates with light-colored backgrounds.

12. Project Difficulty

Since our project involved 3 different models working together, it added to the overall difficulty.

Dealing with the interdependencies between the inputs and outputs of some of our models proved to be a challenge. This was something that we were not familiar with, so it caused a number of problems. If any of the intermediate models, such as character segmentation, failed then the later models, such as character recognition, fail, as incorrect characters are passed, if any. In this way, our system accuracy is dependent on the worst performing model.

13. References

- [1] S. Kaur and S. Kaur, "An Efficient Approach for Number Plate Extraction from Vehicles Image under Image Processing," 2014. Accessed: June 10, 2020. [Online]. Available: <https://pdfs.semanticscholar.org/aa00/bf4d77db46676ba36d4022435887a2b2434b.pdf>
- [2] F. Xie, M. Zhang, J. Zhao, J. Yang, Y. Liu, and X. Yuan, "A Robust License Plate Detection and Character Recognition Algorithm Based on a Combined Feature Extraction Model and BPNN," 2018. Accessed: June 10, 2020. [Online]. Available: <https://www.hindawi.com/journals/jat/2018/6737314/>
- [3] H. Lyu, "Automatic Vehicle Detection and Identification using Visual Features," Windsor, Ontario, Canada, 2018. Accessed: June 10, 2020. [Online]. Available: <https://scholar.uwindsor.ca/cgi/viewcontent.cgi?article=8380&=&context=etd&=&seir edir=1&referer=https%253A%252F%252Fwww.google.com%252Furl%253Fq%253Dhttps%253A%252F%252Fscholar.uwindsor.ca%252Fcgi%252Fviewcontent.cgi%253Farticle%25253D8380%252526context%25253Detd%2526sa%253DD%2526ust%253D1592010037786000%2526usg%253DAFQjCNEZUiG7MGxS4KU5QnbMj2FPrUX4fQ#search=%22https%3A%2F%2Fscholar.uwindsor.ca%2Fcgi%2Fviewcontent.cgi%3Farticle%3D8380%26context%3Detd%22>
- [4] "openalpr/benchmarks", *GitHub*, 2020. [Online]. Available: <https://github.com/openalpr/benchmarks/tree/master/endoend>. [Accessed: 07- Aug- 2020].
- [5] "The Chars74K image dataset - Character Recognition in Natural Images", *Ee.surrey.ac.uk*, 2020. [Online]. Available: <http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/>. [Accessed: 05- Aug- 2020].
- [6] "Cars Dataset," Stanford Artificial Intelligence Laboratory. Accessed: June 11, 2020. [Online]. Available: https://ai.stanford.edu/~jkrause/cars/car_dataset.html
- [7] "Faster R-CNN: Down the rabbit hole of modern object detection | Tryolabs Blog", *Tryolabs.com*, 2020. [Online]. Available: <https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/>. [Accessed: 10- Aug- 2020].
- [8] "Architecture Icons", *Amazon Web Services, Inc.*, 2020. [Online]. Available: <https://aws.amazon.com/architecture/icons/>. [Accessed: 10- Aug- 2020].
- [9] "R for Deep Learning (I): Build Fully Connected Neural Network from Scratch – ParallelR", *Parallelr.com*, 2020. [Online]. Available: <http://www.parallelr.com/r-deep-neural-network-from-scratch/>. [Accessed: 10- Aug- 2020]

Part B: Individual Contribution

1. Overall Contribution Scores

- Deep Pandya - 25%
- Andy Jiang - 25%
- Sophie Kim - 25%
- Johnathon Martin - 25%

2. Individual Contribution Summaries

Task Description	Sophie	Andy	Johnathon	Deep
Stanford Dataset Splitting	25%	75%		
Stanford Dataset Re-labeling		100%		
Data augmentation for Car Type Dataset	100%			
Character Recognition Baseline Model Testing using Amazon Rekognition				100%
Augmenting Character Recognition Dataset			100%	
License plate locator R-CNN Training				100%
Vehicle Type baseline model training & testing	50%	50%		
OpenALPR Dataset Aggregation				100%
Chars74K Dataset Splitting			100%	
Chars74K Dataset Renaming			100%	
Character Segmentation from cropped plates	25%			75%
Tuning hyperparameters on primary CNN for vehicle type classification	50%	50%		
Tuning hyperparameters on primary CNN for character classification			100%	

Experimenting with transfer learning models for vehicle type recognition				100%
Vehicle Type Testing Confusion Matrix	100%			
Character Recognition Testing Confusion Matrix			100%	
Creating a website for demo (React front end + Flask back end)				100%
Packaging components into end to end system (a function that takes in an image and outputs vehicle type and license plate)			100%	
Collecting Test Images from friends	100%			
Collecting Street View Images for vehicle type testing	25%	25%	25%	25%
Creating Final Report	25%	25%	25%	25%
Creating Final Presentation	25%	25%	25%	25%
Recording for presentation	25%	25%	25%	25%

Tasks we were unable to complete

- Perspective transform on license plates
 - The goal was to transform cropped plates into standard rectangles before character segmentation. The results were very inconsistent and as a team, we decided that this was not necessary as character segmentation worked well.
- Real-time license plate and vehicle type extraction from video
 - Was a stretch goal, did not have enough time to attempt