Ontario License Plate Extraction

McMaster University
COMPENG 4TN4 Project
Student:Shuaichen Yan
Supervisor: Seyed Mehdi Ayyoubzadeh

Abstract

In this paper, the extraction of street vehicles license plates is implemented with both machine learning model and deep learning model. Special efforts are devoted and explained here about the pre-process stage, feature extraction stage and classify stage.

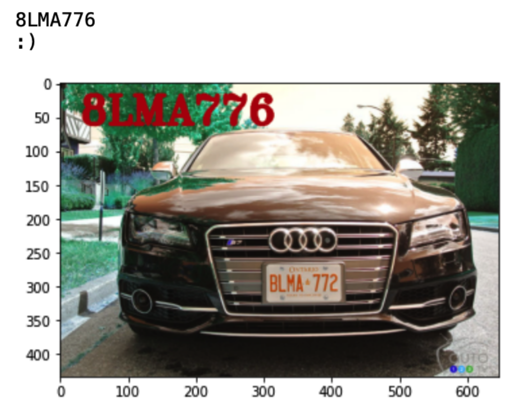

Figure 1. sample result

1. Introduction

Nowadays in the field of computer vision, machine learning is becoming more and more important for sensors to detect and analyze. With strong capability of image processing and extracting gained in the course COMPENG 4TN4 at McMaster University, machine learning models are designed to help predict the characters on local ontario license plates.

It is a python based design project with the applications of tools in cv2 and tensorflow library. The project is coded and designed at the platform of Jupyter Notebook, which gives stable running environment as well as neat visual help.

The choice of the topic is with carefully consideration of the material and support covered by lectures and tutorials. Everything single piece of code would not work correctly without the supportive instructions and assistance from Professor Mehdi as well as his responsible teaching assistances. Finishing the whole project is viewed as a special way to review for the whole semester as well as showing appreciation to great teaching power of this course.

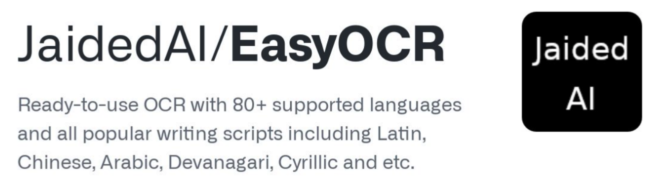

Figure 2. easyOCR

Related work of license plate extraction is introduced as well. It is researched that the library 'easyocr' designed by Jaided AI is commonly used for experiment and research purpose. It is a well-designed library which enables the process of extraction by passing the input image into the function called easyocr.Reader(). It is a mature function which is capable of recognizing multiple languages worldwide while maintaining high detection accuracy. [1]

2. Proposed Method

The project is divided into three stages and implemented with three models.
1. The very first stage is to preprocess the image with the tool of gaussian blur, morphology operation as well as binary image threshold. The edges are found as well so that contours of the license plate as well as individual characters on the plate are located with canny detector.
2. The second stage is the feature extraction which takes features like HOG, center point, width, height, contour area, contour perimeter of the character,etc.
3. The third stage is to train the SVM model with the fea-

tures mentioned above as well as another DL model so that the model is able to classify letters and numbers.

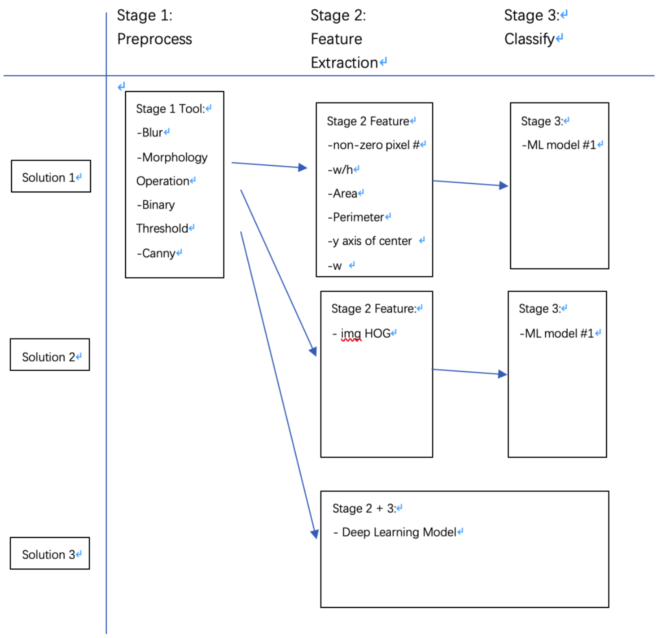

Figure 3. Block Diagram

3. Experimental Results

The main proposed solution, which is solution 1 from the above block diagram is explained with more details here.

3.1. Stage 1 Pre-processing

In this stage, the input image is firstly fed into the program. [3]

Figure 4. Original Image

The license plate is then found by applying the knowledge of canny operator as well as rectangle contour finding.

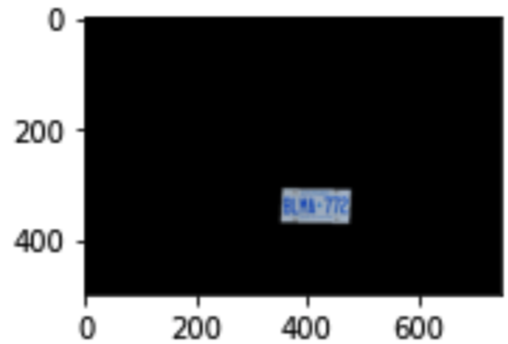

Figure 5. absolute location of plate

The license plate is then saved to be further processed.

Figure 6. saved plate

The plate is then blurred, opened and converted to binary image.

Figure 7. preprocess

The cv2.findContour() function is used again to extract characters.

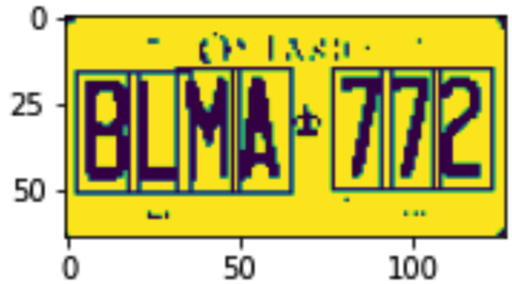

Figure 8. find character

The characters found are then saved to the working directory. They are all with size of 18 x 35.

Figure 9. saved characters

Figure 10. characters in working dir

The characters are then resized to square 20 x 20 for model.

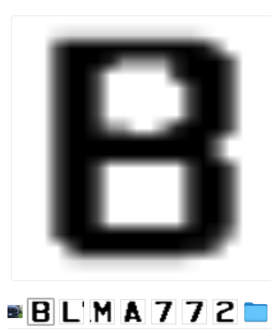

Figure 11. resized image

Figure 12. resized characters in working dir

3.2. Stage 2: Feature Extraction

Here in this stage, the 6 features of characters both from dataset(for model traing and testing) and user picked data for project (for pure predicting) are picked and visualized to see their mean and variance. The 6 features are [non-zero pixel , w/h, Area, Perimeter, y axis of center, width]

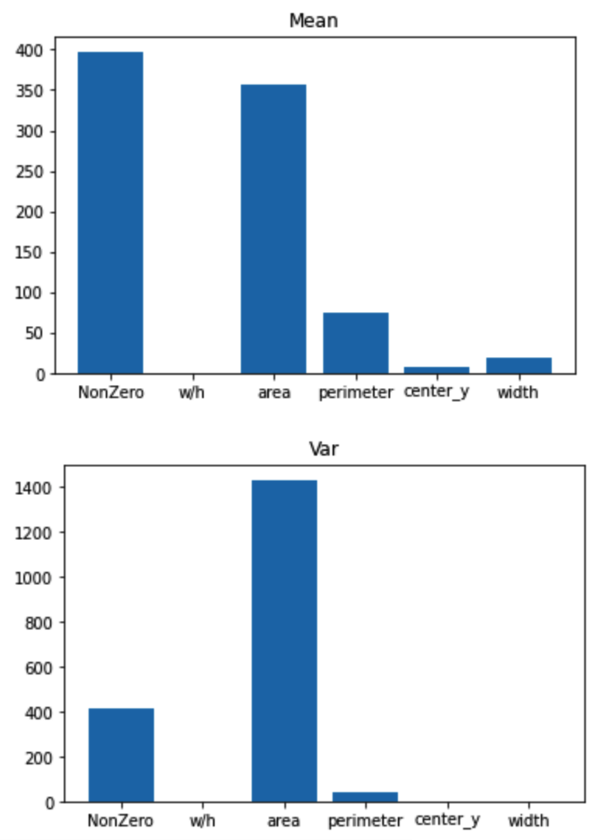

Figure 13. features extracted

The features achieved are then split into training set and testing set. The model is trained based on the training set with extracted features considered. The test set is then fed to the SVM model to predict.
To compare the model quality, another machine learning model is trained with HOG of the image as the extracted features. HOG is another way to describe image by considering the histogram of the gradients of every single pixels in the local area so that the edges and other features can be described. Then the histogram, in the form of a numpy ar-

ray, is passed to the model to be trained. The same training set and testing set are used to find model quality. [4]

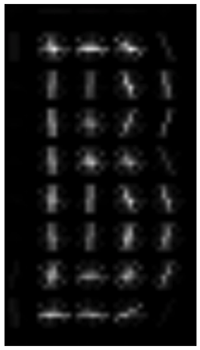

Figure 14. HOG of letter B

Figure 15. Letter B

Before model results are provided, some keywords used to describe the quality of the model is explained here.

$$\text{Precision} = \frac{\textit{True Positive}}{\textit{True Positive} + \textit{False Positive}}$$

$$= \frac{\textit{True Positive}}{\textit{Total Predicted Positive}}$$

$$\text{Recall} = \frac{\textit{True Positive}}{\textit{True Positive} + \textit{False Negative}}$$

$$= \frac{\textit{True Positive}}{\textit{Total Actual Positive}}$$

$$\text{F1} = 2 \times \frac{\textit{Precision} * \textit{Recall}}{\textit{Precision} + \textit{Recall}}$$

By analyzing the equations above, it is found that while recall calculates how many of the actual positives the model capture through labeling it as positive, precision talks about how precise/accurate the model is out of those predicted positive. [2]

0.9556338028169014

	precision	recall	f1-score	support
N	0.97	0.78	0.87	50
C	0.91	0.98	0.94	43
J	0.91	0.98	0.94	43
X	0.95	1.00	0.97	37
H	1.00	0.97	0.99	40
E	1.00	1.00	1.00	38
8	0.92	0.92	0.92	39
3	0.98	0.94	0.96	47
2	0.90	0.95	0.93	40
6	1.00	1.00	1.00	48
5	0.99	1.00	0.99	77
Q	0.88	0.95	0.91	44
R	0.89	0.98	0.93	42
7	0.96	0.89	0.93	28
F	0.95	0.86	0.90	49
P	0.85	0.98	0.91	42
L	0.93	0.91	0.92	45
9	0.98	0.95	0.96	42
B	0.98	0.96	0.97	52
Z	0.98	1.00	0.99	58
A	1.00	1.00	1.00	41
S	1.00	1.00	1.00	45
1	0.96	0.93	0.95	46
V	0.97	0.95	0.96	38
G	0.87	1.00	0.93	41
4	1.00	0.88	0.94	34
0	1.00	1.00	1.00	38
T	0.89	0.89	0.89	38
Y	1.00	0.97	0.99	34
M	0.96	0.96	0.96	45
U	0.98	0.96	0.97	45
D	1.00	0.97	0.99	38
K	1.00	1.00	1.00	33
accuracy			0.96	1420
macro avg	0.96	0.95	0.95	1420
weighted avg	0.96	0.96	0.96	1420

Figure 16. accuracy for model with 6 feature

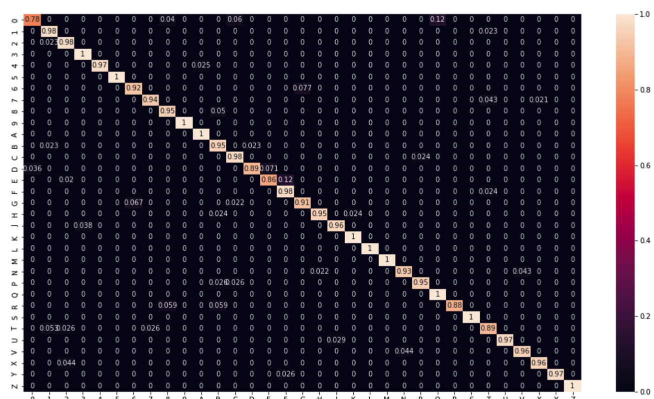

Figure 17. confusion matrix for model with 6 feature

After feeding the 6 feature model (main approach) with the testing dataset, the model is demonstrated to be good at detection most of the characters.
The testing result of model 2 is analyzed as well.

```
0.9394366197183098
              precision    recall  f1-score   support

           N       0.92      0.66      0.77        50
           C       0.85      0.95      0.90        43
           J       0.91      0.95      0.93        43
           X       0.97      1.00      0.99        37
           H       0.95      0.97      0.96        40
           E       0.97      0.97      0.97        38
           8       0.88      0.90      0.89        39
           3       1.00      0.96      0.98        47
           2       0.90      0.95      0.93        40
           6       0.96      0.98      0.97        48
           5       0.95      0.96      0.95        77
           Q       0.90      0.86      0.88        44
           R       0.89      0.95      0.92        42
           7       0.74      0.89      0.81        28
           F       0.96      0.88      0.91        49
           P       0.89      1.00      0.94        42
           L       0.80      0.91      0.85        45
           9       0.90      0.88      0.89        42
           B       1.00      0.96      0.98        52
           Z       0.98      0.98      0.98        58
           A       0.98      1.00      0.99        41
           S       0.98      0.96      0.97        45
           1       1.00      0.96      0.98        46
           V       0.97      0.95      0.96        38
           G       0.93      0.98      0.95        41
           4       1.00      0.94      0.97        34
           0       0.97      0.97      0.97        38
           T       1.00      0.87      0.93        38
           Y       0.86      0.91      0.89        34
           M       0.98      1.00      0.99        45
           U       1.00      0.96      0.98        45
           D       1.00      0.95      0.97        38
           K       1.00      1.00      1.00        33

    accuracy                           0.94      1420
   macro avg       0.94      0.94      0.94      1420
weighted avg       0.94      0.94      0.94      1420
```

Figure 18. accuracy for HOG feature

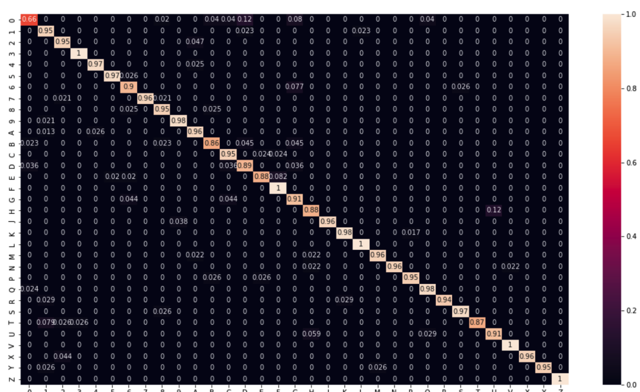

Figure 19. confusion matrix for HOG feature

By comparing the stats of the two models, it is realized that the first ML model, the model with the training on 6 features of [non-zero pixel , w/h, Area, Perimeter, y axis of center, width] is giving better accuracy.

3.3. Stage 3: Classify

In this stage of the project, the classification result is achieved by feeding target license plates in Ontario into the ML model so that the result is predicted.

```
#PREDICT MY RESULT
y_pred_my_feature = model_linear2.predict(My_FEATURE)
print(y_pred_my_feature)
my_string2=''.join(y_pred_my_feature)
print(my_string2)

['4' 'L' 'M' 'A' '2' '2' '4']
4LMA224
```

Figure 20. predict with model with 6 feature

```
['4' 'L' 'M' 'A' '2' '2' '4']
4LMA224
```

Figure 21. result with 6 feature model

```
#PREDICT MY RESULT
y_pred_my = model_linear.predict(My_HOG)
print(y_pred_my)

['8' 'L' 'M' 'A' '7' '7' '6']
```

Figure 22. predict with model with HOG feature

```
8LMA776
:)
```

Figure 23. result with HOG model

By comparing the detection results on target Ontario license plate image with plate ’BLMA772’, model based on HOG feature gives the prediction of ’8LMA776’, while the model based on 6 features gives the prediction of '4LMA224'. It is realized that although HOG feature model gives lower accuracy with testing set, it gives better detection of 5/7 corrected with user defined image. The 6 feature model, on the other hand, gives a higher testing accuracy while only gives 3/7 corrected.

4. Deep Learning Method

The third approach to address the license plate challenge is documented here. The training set as well as the testing set are imported first.

```
Found 864 images belonging to 36 classes.
Found 216 images belonging to 36 classes.
```

Figure 24. import data

The Deep learning model is then designed.

```
Model: "sequential_35"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_64 (Conv2D)          (None, 20, 20, 32)        55328

 max_pooling2d_57 (MaxPoolin  (None, 10, 10, 32)       0
 g2D)

 dropout_6 (Dropout)         (None, 10, 10, 32)        0

 flatten_28 (Flatten)        (None, 3200)              0

 dense_72 (Dense)            (None, 128)               409728

 dense_73 (Dense)            (None, 36)                4644

=================================================================
Total params: 469,700
Trainable params: 469,700
Non-trainable params: 0
_________________________________________________________________
```

Figure 25. model summary

The model is then trained with 15 epoch due to the fact that the user laptop does not have GPU for more epoch run.

```
Epoch 11/15
864/864 [==============================] - 26s 30ms/step - loss: 1.2807 - accuracy: 0.6620
Epoch 12/15
864/864 [==============================] - 27s 31ms/step - loss: 1.1809 - accuracy: 0.6921
Epoch 13/15
864/864 [==============================] - 28s 32ms/step - loss: 1.0550 - accuracy: 0.7188
Epoch 14/15
864/864 [==============================] - 27s 32ms/step - loss: 1.0090 - accuracy: 0.7338
Epoch 15/15
864/864 [==============================] - 27s 31ms/step - loss: 0.9950 - accuracy: 0.7280

<keras.callbacks.History at 0x16c1931c0>
```

Figure 26. 15 epoch accuracy

After getting the DL model ready, the preprocessed characters, which are the same user defined image with the above two ML models, are feed into DL model.

Figure 27. user defined set

```
['B', 'L', '3', 'A', '7', '7', 'N']
BL3A77N
```

Figure 28. DL model prediction

The result is then shown above. It is realized that the deep learning model manages to detect 5/7 characters from the plate.

5. Conclusion

In this design project, two machine learning models and one deep learning model are designed to recognize Ontario license plate. The predicting process involves the stage of pre-processing, fearure extracting as well as classifying. The testing accuracy of the models are promising with accuracy over 90 percent. The capability to recognize license plate is ranged from 37.5 percent to 75 percent among three models.

The accuracy of the models can be further improved by:
1. Train with bigger data set
2. Design more robust ML model
3. Run DL model with more epoch

References

[1] Adrian Rosebrock. Getting started with easy-ocr for optical character recognition, 2020. https://pyimagesearch.com/2020/09/14/getting-started-with-easyocr-for-optical-character-recognition/. 1

[2] Koo Ping Shung. Accuracy, precision, recall or f1?, 2018. https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9. 4

[3] Alexandra Straub. 2013 audi s7 4.0 tfsi quattro review, 2013. http://www.auto-venus.com/en/2013-audi-s7-40-tfsi-quattro-review/1709/. 2

[4] Mrinal Tyagi. Hog (histogram of oriented gradients): An overview, 2021. https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f. 4