Research on the Optimal Machine Learning Classifier for Traffic Signs

Boyu Wang*

University of Macau, Faculty of Science and Technology, Macau, 999078, China

ABSTRACT: Now autonomous driving is a hot topic, and the identification of traffic signs is also extremely important for autonomous driving. This paper mainly compares the difference of the Support Vector Machine (SVM), Multilayer Perceptron (MLP), and Logistic Regression (LR) Classifier in the traffic sign classification. The effect of the initial image processing on classification accuracy is also studied. The paper found that sharpening the image significantly improved the accuracy of the image classification. Based on the results of various situations, the author found that, in this paper, SVM is the classifier with the best classification effect, but the effect of LR classifier is not much worse than that of SVM when the image is sharpened.

1. INTRODUCTION

Research suggests that by 2030, the annual passenger traffic will increase by 50% while the global freight volumes will grow by 70%; another 1.2 billion cars will travel in our streets by 2050 [1], so in the future, the autonomous driving will become increasingly important. In this paper, the author mainly intends to study the different machine learning classifiers for the traffic sign classification and select the optimal one. At the same time, python is used as the programming language for the training and testing of each machine learning classifier. This paper mainly discusses the Support Vector Machine, Multi-layer Perceptron and Logistic Regression classifier, and finds out the parameter settings of each classifier for the traffic sign classification. Moreover, the data processing method that can improve the success rate of the classifier is applied in the study. The influence of different machine learning classifiers on the image acquisition under different circumstances (for example, the image brightness obtained in the dark is too low) is comprehensively analyzed to obtain a better classifier. This paper's results can provide some help for the recognition of the traffic signs by machines, thus further promoting the development of autonomous driving.

2. METHODOLOGY

2.1. Data selection

The data used in this paper is the PPM (Portable PixMap) file. The PPM format is a lowest common denominator color image file format. It is relatively straightforward to write programs capable of processing the PPM format, so this paper chooses this type of picture file for research. However, their lack of compression means that PPM files can be big, so they are not ideal for storing large-scale images [2]. Therefore, the author chooses traffic sign images without particularly high resolution for research. Moreover, most of the traffic sign images are not too large, which controls the size of the image files.

For the preliminary study, fifteen different traffic signs were selected and assigned the labels from Class0 to Class14 (seen in Figure 1). The training data is classified at the beginning so that the data of the same class is in the same folder and has the same label. When designing the input for the model, the original data and the labels correspond well. Then when training the classifier, the corresponded data and labels are given to the classifier, and the classifier will automatically process multiple classes. After the training is completed, the classifier will also classify the same number of classes for the new test data. This will facilitate the later check of whether the classifier results are correct, and thus calculating the accuracy rate.

Figure 1 Class0 and class14.

2.2. Data pre-processing

The pre-processing of the data is mainly to process the image. In this study, the image binarization and image

*Corresponding author. Email: db92823@umac.mo

enhancement (image sharpening) are mainly used. The image binarization can reduce the amount of data in the image, and make the contour of the target stand out. Image enhancement can be used to provide better input for other automated image processing systems [3]. In order to perform image pre-processing, the OpenCV-Python function package is introduced into the data processing part of the source code.

Image binarization is to set the gray value of the pixels whose gray value is greater than or equal to the threshold value to 255 and set the gray value of the rest to 0. The two algorithms in Figure 2 below can realize the image binarization, and the two algorithms were used for training and testing twice in the study. The obtained machine learning classifiers all had success rates of around 95% (The specific value is 0.9493043030146869. LR classifier is used for testing), which did not make much difference to the results produced by the unprocessed images. So the binarization of the image to be classified will not improve the success rate of the classifier.

```
cvThreshold( dst, dst,230 , 255, CV_THRESH_BINARY_INV);
cvAdaptiveThreshold( dst, dst, 255, CV_ADAPTIVE_THRESH_MEAN_C,CV_THRESH_BINARY, 9, -10);
```

Figure 2 Two algorithms of the image binarization.

In the image enhancement, sometimes in order to emphasize the edges and details of the image, it is necessary to sharpen the image to improve the contrast. The main goal of the image sharpening is to enhance the image and make the image appear clearer and brighter [4]. In this paper, the Laplacian operator is mainly used as the second derivative for the image sharpening. The Laplacian can be defined as the following function f(x,y) (Figure 3) by two variables x, y. Through changing this function to another expression, and using x, y as the coordinate center point, it can be expressed as Figure 4. And as shown in Figure 5, the original image and the Laplacian image are superimposed to sharpen the convolution kernel template. The data in this study is sharpened using the method above, and the source code is shown in Figure 6. In separate tests of the three classifiers, the sharpened images are better for the classifier to distinguish different categories of images, and the accuracy is improved by 2% compared with the original one.

$$\nabla^2 f = \frac{\partial^2 f}{\partial^2 x} + \frac{\partial^2 f}{\partial^2 y}$$

$$\frac{\partial^2 f}{\partial^2 x} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f}{\partial^2 y} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\nabla^2 f = \frac{\partial^2 f}{\partial^2 x} + \frac{\partial^2 f}{\partial^2 y} = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

Figure 3 The Laplacian function.

0	1	0
1	-4	1
0	1	0

Figure 4 Laplacian function special expression.

0	-1	0
-1	5	-1
0	-1	0

Figure 5 Sharpen the convolution kernel template.

```
img=img.resize((32,32), Image.BICUBIC)
kernel = np.array([[0, -1, 0], [-1, 5, -1], [0, -1, 0]], np.float32)
img=np.array(img)
img = cv2.filter2D(img, -1, kernel=kernel)
```

Figure 6 Sharpen part of the source code.

2.3. Three different classifiers

2.3.1. Logistic Regression classifier

Logistic Regression is a classification technique used in machine learning. It uses a logistic function to model the dependent variable [5]. The Logistic Regression can be divided into three main types. The first is the Binary Logistic Regression, which classifies objects into only two types; the second is the Multinomial Logistic Regression, which classifies objects into three or more types; and the last is the Ordinal Logistic Regression, which classifies objects into ordered types.

In this paper, the Multinomial Logistic Regression is used, which is different from the binary classification in that it outputs a probability distribution representing the probability of each class. The Multinomial Logistic Regression uses the Softmax function to map the weighted summation of features into a probability distribution. The SoftMax function can be represented in Figure 7. The x represents the classification object, and its probability of belonging to the category r is P, and this function satisfies $\sum_{r=1}^{K} p(r|\vec{x_i}) = 1$ for all categories r. An important

characteristic of the parameters of the logistic regression is the existence and consistency of the maximum likelihood parameters [6]. Then estimate and establish the likelihood function expression (Figure 8) by the maximum likelihood method, and finally obtain the loss function (Figure 9) by the logarithmic operations.

$$P(r \mid \vec{x}_i) = \frac{\exp(\vec{w}_r \cdot \vec{x}_i)}{\sum_{j=1}^{K} \exp(\vec{w}_j \cdot \vec{x}_i)}$$

Figure 7 SoftMax function.

$$L = \prod_{i=1}^{n} P(y_i \mid \vec{x}_i)$$

Figure 8 Likelihood function expression.

$$\begin{aligned} logL &= -\sum_{i=1}^{n} \log[P(y_i \mid \vec{x}_i)] \\ &= \sum_{i=1}^{n} \underbrace{\{-\vec{w}_{y_i} \cdot \vec{x}_i + \log \sum_{j=1}^{K} \exp(\vec{W}_j \cdot \vec{x}_i)\}}_{\log L_i} \end{aligned}$$

Figure 9 Loss function.

For parameter settings for the logistic regression, there are there fourteen parameter types. In this paper, the parameter c (the reciprocal of the regularization coefficient λ, float type, and the default is 1.0, must be a positive floating point number, smaller values indicate stronger regularization) is mainly used. The parameter sag (stochastic average gradient descent is a variant of the gradient descent method. The difference from the ordinary gradient descent method is that each iteration only uses a part of the sample to calculate the gradient, which is suitable when there is a lot of sample data) is also used. As seen in Figure 10, the regularization strength is the default of 1, so c is set to 1. And because the traffic sign datasets contain a large amount of historical data, the solver is set to 'sag'.

```
logreg = LogisticRegression(C=1, solver="sag")
```

Figure 10 Source code of setting the LR parameter.

2.3.2. Multilayer Perceptron

Neural networks are a common problem in the current machine learning field and the main problem solved by neural networks is classification. In this paper, a multilayer perceptron is used, which is a forward-structured artificial neural network that maps a set of input vectors to a set of output vectors. MLP has many applications in computer vision, and MLP is well suited for the pattern classification. Thus, the majority of the MLP applications in the field of image analysis have been in the area of image/object recognition [7].

As mentioned above, the MLP is selected as a traffic sign classifier for comparison in this paper. The neural network is derived from the study and the simulation of biological neurons, which are mainly composed of dendrites, cell bodies, and axons. The dendrite is the input terminal of the cell body, and the axon is the output terminal of the cell body. The structure of MLP can be obtained through the biological neuron model. The most common MLP is a three-layer structure: input layer, hidden layer and output layer. The MLP is described as being fully connected, with each node connected to every node in the next and previous layer [8].

As for the activation function for the MLP, i is set as the input layer node, j as the hidden layer neuron node, and k as the output layer node. wij represents the weight of the neuron j, and wjk as the weight of the neuron k. Here the forward propagation and the weighted summation of h are considered to get Figure 11. hj represents the weighted sum of all current inputs. The output value of the hidden layer neuron is shown in Figure 12, and g() represents the activation function. The output value of the output layer is shown in Figure 13. The activation function sigmoid function can be used in the MLP. For the loss function E, the sum of squared errors is used in Figure 15.

$$h_j = \sum_{i=0}^{M} w_{ij} x_j$$

Figure 11 Weighted summation of h.

$$a_j = g(h_j) = g(\sum_{i=0}^{M} w_{ij} x_{ij})$$

Figure 12 Output value of the hidden layer neuron.

$$y = a_k = g(h_k) = g(\sum_{i=0}^{M} w_{jk} x_{jk})$$

Figure 13 Output value of the output layer.

$$g(h) = \sigma(h) = \frac{1}{1 + e^{-h}}$$

Figure 14 Activation function.

$$E = \frac{1}{2} \sum_{k=1}^{N} (y - t)^2$$

Figure 15 The sum of squared errors.

For parameter setting, the Incremental algorithms and Decremental algorithms [9] are mainly used in the selection of the neuron number in this paper. Both algorithms are gradient descent optimization algorithms so there is a need to account for the possibility of reaching

local minima at the beginning. In the use of the algorithm, it is found that for the Decremental algorithms, if the number of neurons set at the beginning is too high, the test training time will be too long, so the final choice is to use the Incremental algorithms, starting from 10 neurons and increasing by five neurons each time. After many comparison experiments, it is found that the number of hidden layers is set to 1 and the number of neurons is set to 50 to achieve the best results for the data set of this study and this will not cause the training time to be too long. The source code for parameter settings can be seen in Figure 16.

```
mlp = MLPClassifier(hidden_layer_sizes=(50, ), max_iter=10, alpha=1e-4,
                    solver='sgd', verbose=10, random_state=1, learning_rate_init=.1)
```

Figure 16 Source code for MLP parameter settings.

2.3.3. Support Vector Machine

SVM is a data mining method based on statistical learning theory, which can successfully deal with the regression problems (time series analysis) and pattern recognition (classification problems, discriminant analysis) and many other problems [10]. SVM is a generalized linear classifier for binary classification of data in a supervised learning manner. Its principle is to find an optimal classification hyperplane that meets the classification requirements, so that the hyperplane can maximize the blank areas on both sides of the hyperplane while ensuring the classification accuracy.

SVM itself is a binary classifier, it can also deal with multi-classification problems. When dealing with multi-classification problems, it is necessary to construct a multi-class classifier. There are two main methods for constructing SVM multi-classifiers, one is the direct method, and the other is the indirect method.

The direct method seems simple, but its computational complexity is high, and it is difficult to implement. It is more suitable for dealing with small problems, so in this paper, the one-vs-the-rest method is selected in the indirect method. SVM mainly has three methods: LinearSVC, NuSVC and SVC. LinearSVC is used in the paper. A Linear SVC aims to match the data you have and return the "right fit" hyperplane, which separates or classifies the information [11]. For LinearSVC, a total of twelve parameters can be set: ‘penalty’ regularization parameter, ‘oss’ loss function, ‘tol’ residual convergence condition, ‘C’ penalty coefficient, ‘mutli_class’ classification strategy specification, ‘fit_intercept’ whether to calculate the intercept, ‘class_weigt’, whether 'verbose' is redundant, ‘random_state’ random seed size, ‘max_iter’ maximum number of iterations, ‘intercept_scaling’ and ‘dual’ [12]. The source code of SVM parameter setting in this paper is shown in Figure 17.

```
lin_clf = svm.LinearSVC(C=1)
```

Figure 17 Source code of SVM parameter setting.

3. RESULT AND DISCUSSION

3.1. Images that are hard to be classified

Images class7 and class8 (speed limit 100 speed limit 120) are two classes that are difficult to distinguish because there are many speed-limited labels, and they are two of the more similar classes. When they have low brightness and low clarity, it is difficult for the classifier to classify them. The following figure shows the occurrence of the SVM on the test set according to the label. As shown in Figure 18, in image classification, these 8 images are classified and labels are printed to show the classification results.

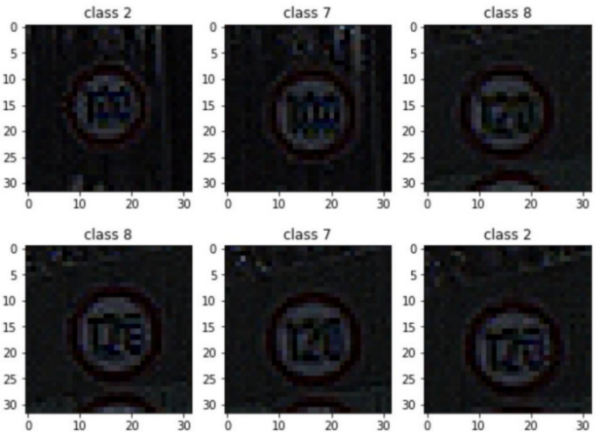

Figure 18 Classification results of eight images.

3.2. Performance of the three classifiers

```
Logistic final Classification Accuracy:
Out[77]: 0.9775280898876404
```

Figure 19 Accuracy of Logistic Regression classifier.

```
SVM final Classification Accuracy:
Out[81]: 0.9887640449438202
```

Figure 20 Accuracy of SVM classifier.

```
MLP final Classification Accuracy:
Out[79]: 0.9550561797752809
```

Figure 21 Accuracy of MLP classifier.

As seen from the above Figure 19-21, SVM is the best performing classifier, followed by logistic regression and finally MLP.

Finally, cross validation is used in this paper. In the cross-validation stage, the cross-validation results of the three classifiers are printed out using the cross_val_score function (source code in Figure 22). Cross-validation is a

statistical method used to estimate the skill of the machine learning models. It is used to verify that the results are accurate.

```
from sklearn.model_selection import cross_val_score

scores1 = cross_val_score(logreg, X_train, Y_train, cv=5)
scores2 = cross_val_score(lin_clf, X_train, Y_train, cv=5)
scores3 = cross_val_score(mlp, X_train, Y_train, cv=5)

print("Logistic Accuracy: %0.2f (+/- %0.2f)" % (scores1.mean(), scores1.std() * 2))
print("SVM Accuracy: %0.2f (+/- %0.2f)" % (scores2.mean(), scores2.std() * 2))
print("MLP Accuracy: %0.2f (+/- %0.2f)" % (scores3.mean(), scores3.std() * 2))
```

Figure 22 Source code of the cross-validation.

```
Logistic Accuracy: 0.94 (+/- 0.01)
SVM Accuracy: 0.92 (+/- 0.01)
MLP Accuracy: 0.71 (+/- 0.33)
```

Figure 23 Cross-validation results.

The above Figure 23 is the cross-validation result. Although the cross-validation result is that logistic regression performs the best, SVM always produces stable and accurate test results during testing. And cross-validation is only to verify the reliability of the previous data and does not represent the final result. It can be seen from the cross-validation results that there is little difference between SVM and LR, which proves that the previous results are reliable, so SVM is still the optimal classifier in this paper.

It is interesting to note that without sharpening the images, SVM still performs well, with about 98% accuracy, while logistic regression and MLP give much worse accuracy than SVM. After sharpening, the differences in the images are further amplified and logistic regression also has a quite good correct rate (99.2%), only a little worse than SVM (99.8%). And the accuracy of MLP is also close to 99%. So the performance of logistic regression can be well represented for data with greater differences, but SVM performs well in both cases, which is one of the reasons why SVM is the optimal classifier in this paper. However, MLP performs average in either case, and after the image is processed differently, it is needed to do another processing on the input X value to make the loss lower, which is a rather inconvenient point for MLP. There may be other reasons for the lower accuracy of MLP. For small classes and attributes, it is noted that MLP achieves the worst accuracy [13]. So maybe the attributes of the traffic signs are too few or the classes of the signs are too small.

4. CONCLUSION

This paper mainly compares the differences between SVM, MLP and LR classifiers in the traffic sign classification. The effect of the initial image processing on classification accuracy is investigated. This paper finds that sharpening images can significantly improve the accuracy of image classification. Combining multiple situations and results, the author found that SVM is the best classifier in this paper, and the LR classifier is not much worse than SVM in the image sharpening. The classification effect of MLP is poor, and the accuracy of MLP may be improved by adding traffic sign categories. None of the three classifiers are effective enough in classifying the images with lower brightness.

The training set (more than 15,000 pictures) samples used in this study may still be quite different from the real situation encountered by the autonomous driving car in reality, so the results obtained may be different from the actual driving situation of the car. There are biases, so future research should collect more pictures of the traffic signs in different situations to train and compare the classifiers. To see more source code, please visit: https://github.com/BarryMonkingWang/Comparison-of-SVM-MLP-and-LR-classifier.git.

REFERENCES

1. F. Arena, D. Ticali, The development of autonomous driving vehicles in Tomorrow's Smart Cities Mobility, AIP Publishing, 2018, Retrieved March 1, 2022, from https://aip.scitation.org/doi/abs/10.1063/1.5079196.
2. What are PPM files and how do you open them? | adobe. (n.d.). Retrieved March 2, 2022, from https://www.adobe.com/creativecloud/file-types/image/raster/ppm-file.html.
3. H. Lbrahim, N.S.P. Kong, Image sharpening using sub-regions histogram equalization, IEEE Xplore, 2009, Retrieved March 3, 2022, from https://ieeexplore.ieee.org/abstract/document/5174471.
4. Z. Afrose, Y. Shen, Mesh color sharpening, Advances in Engineering Software, 2015, Retrieved March 4, 2022, from https://www.sciencedirect.com/science/article/pii/S0965997815001398.
5. A. Raj, The perfect recipe for classification using logistic regression, Medium, 2021, Retrieved March 5, 2022, from https://towardsdatascience.com/the-perfect-recipe-for-classification-using-logistic-regression-f8648e267592.
6. A.I. Schein, L.H. Ungar, Active learning for logistic regression: An evaluation-machine learning, SpringerLink, 2017, Retrieved March 5, 2022, from https://link.springer.com/article/10.1007/s10994-007-5019-5.
7. A. Khotanzad, C. Chung, Application of multi-layer Perceptron Neural Networks to vision problems-neural computing and applications, SpringerLink, 1998, Retrieved March 6, 2022, from https://link.springer.com/article/10.1007/BF01414886.
8. M.W. Gardner, S.R. Dorling, Artificial Neural Networks (the multilayer perceptron) - a review of applications in the Atmospheric Sciences, Atmospheric Environment, 1998, Retrieved March 6, 2022, from https://www.sciencedirect.com/science/article/pii/S1352231097004470.

9. P.A. Castillo, J. Carpio, J.J. Merelo, A. Prieto, V. Rivas, G. Romero, Evolving multilayer perceptrons-neural processing letters, SpringerLink, 2000, Retrieved March 6, 2022, from https://link.springer.com/article/10.1023/A:1009684907680.
10. D. Shi-fei, Q. Bing-juan, T. Hong-yan, An Overview on Theory and Algorithm of Support Vector Machines, Journal of University of Electronic Science and Technology of China, 2011, Retrieved March 6, 2022, from https://www.airitilibrary.com/Publication/alDetailedMesh?docid=10010548-201101-201107110024-201107110024-2-10.
11. S.S. Sikarwar, N. Tiwari, Analysis The Sentiments Of Amazon Reviews Dataset By Using Linear SVC And Voting Classifier, 2020, Retrieved Mar. 2022, https://www.researchgate.net/profile/Sandeep-Sikarwar/publication/356727150_Analysis_The_Sentiments_Of_Amazon_Reviews_Dataset_By_Using_Linear_SVC_And_Voting_Classifier/links/61a8dc8aaade5b1bf5fb71cd/Analysis-The-Sentiments-Of-Amazon-Reviews-Dataset-By-Using-Linear-SVC-And-Voting-Classifier.pdf.
12. Sklearn.svm.LinearSVC. scikit. (n.d.). Retrieved March 7, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html.
13. E.A. Zanaty, Support Vector Machines (svms) versus multilayer perception (MLP) in Data Classification, Egyptian Informatics Journal, 2012, Retrieved March 8, 2022, from https://www.sciencedirect.com/science/article/pii/S1110866512000345.