XI'AN JIAOTONG-LIVERPOOL UNIVERSITY
西 交 利 物 浦 大 学
YEAR 4
COURSE WORK SUBMISSION

Name	Yuxuan	Wu
ID Number	1716309	
Programme	INT301	
Module Title	Biocomputation	
Module Code	INT301	
Assignment Title	Assessment2	
Submission Deadline	Time of submission:12.20.2020	
Lecturer Responsible	Rui Yang	

Signature ……. …吴宇轩……………………… Date …………………1.6.2021……………..

For Academic Office use:	Date Received	Days Late	Penalty

Introduction

Image classification, one of the typical applications of a neural network, has gained significant importance and is widely implemented in recent years. Therefore, in this project, I would design five algorithms to classify the segmented characters from the automobiles’ license plates. The algorithms used in this project could be multi-layer perceptron (MLP), convolutional neural network (CNN), learning vector quantization (LVQ) and two radial basis function (RBF) networks with k-means returned center or self-organizing map returned center. Additionally, necessary preprocessing and normalization were included to ensure the quality and performance of the model. Furthermore, each model's performance could be compared with the accuracy and the confusion matrix.

Methodology

Data preparation

The provided dataset is a zip file containing segmented characters from the license plates of automobiles. Specifically, the dataset contains 24 symbols with 100 samples in each category. Owing to the existence of different formats, either jpeg or jpg, and the chaos naming styles on each sample, I decided to rewrite all the samples into one file (ass2_professed_data) with uniform styles, for instance, “A1.jpeg”. Therefore, the newly created file folder with proper names has 2400 samples and prepare to conduct further analysis.

Once unified the raw data, I have to read the input figure (48*24) and transform it into a 1D numerical matrix (1*1152) iteratively. Furthermore, the normalization method should be incorporated to reduce the number range into 0 to 1, hoping to minimize the potential outliers’ influence. In addition, one-hot encoding was utilized to transfer each label into a 1*24 vector. For instance, the character "A" could be transformed into 1. Moreover, the existing driving license does not contain the character "I" or "O", so the numbers are assigned to characters accordingly. For training and testing, I split the dataset into a training set (80%) and a testing set (20%). Moreover, the preprocessed data were then saved for following model training (“train_test_data.mat”), except for CNN.

Regarding the CNN, I utilized “imageDataset” to store and process the input figure directly from the provided dataset since the function used in CNN is a highly assembled one and devoid of previous preprocessing. Additionally, the model training process of CNN differs from other models, with each figure (48*24) directly fed into the model. As a result, the method would be different in the model of CNN. Similarly, the dataset was divided into 8:2, with training samples accounting for 80% of the dataset.

Multi-layer Perceptron (MLP)

MLP classifier is a classical artificial neural network (ANN) widely implemented in classification problems. Generally, the simple network is composed of three components: input layer, hidden layer and output layer, and with the back-propagation training algorithms to update the weights until convergence [1]. In this case, the MLP model contains three hidden layers with hyperbolic tangent sigmoid as the activation function (Figure 1). As previously mentioned, each figure was compressed into a 1D matrix with 1152 features, so the input features would be 1152. Regarding to the output layer, I used the linear activation function for simplicity, and the output layer could produce probabilities for 24 clusters. Hyperparameters involved in this MLP are summarized in Table 1.

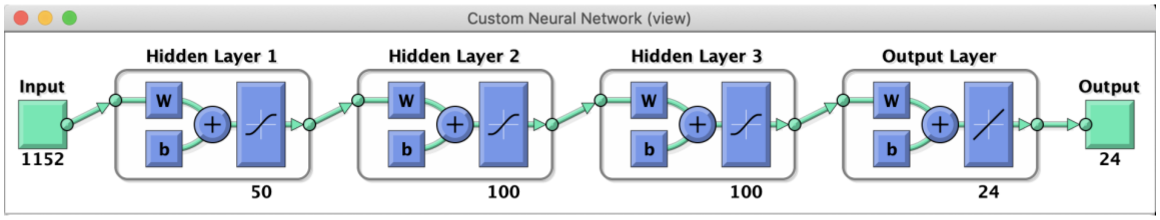


Figure 1: Components of custom MLP

Epoch	Learning rate	Momentum	Neurons in each layer
1000	0.2	0.4	[50,100,100,24]

Table 1: Hyperparameter setting in MLP

Convolutional Neural Network (CNN)

CNN is a deep learning algorithm widely used in image classification [2]. In this experiment, three hidden blocks with one fully connected layer were devised to build the network and stochastic gradient descent with momentum (SGDM) was utilized to update the parameters. Each figure with size 48*24 was fed into the model, the input layer thus set to (48,24,1). Regarding the three hidden blocks, most components were shared, except for the number of filters, the number was set to 8, 16, 32 for each block. In each convolutional block, kernel size was set to 3*3 and the "same" padding technique was utilized in the convolutional layer. Batch normalization was followed to eliminate the influence of outliers. Additionally, Rectified Liner Unit (ReLU) was chosen as the activation function and the following Max pooling layer set both size and stride equal 2. Finally, a fully connected layer with 24 neurons and softmax as activation function (Figure 2). The probability of each label was calculated, and the highest score could be considered as the predicted label.

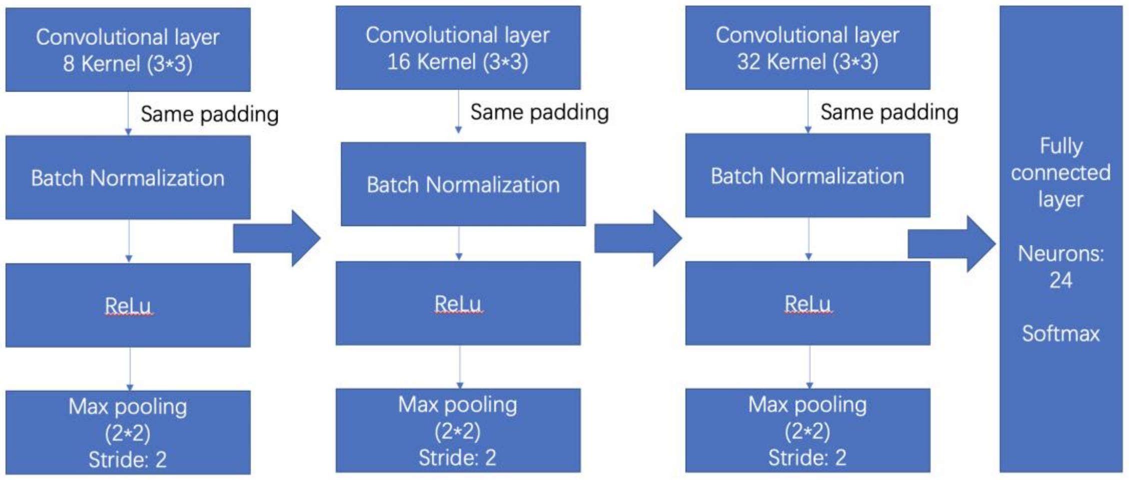


Figure 2: Components of custom CNN

Learning Vector quantization (LVQ)

LVQ is a supervised learning technique with a collection of codebook vectors, dividing the training dataset into different sections [3]. Predictions are made for a new instance by searching through all codebook vectors for the most similar instances and moving the centers slightly. To be more specific, it applies a winner-take-all approach and belongs to the competitive learning network domain. Normally, Euclidean Distance would be used to determine the similarity between sections. In this method, 360 clusters were devised to train the LVQ network. Figure 3 is the structure of the LVQ network.

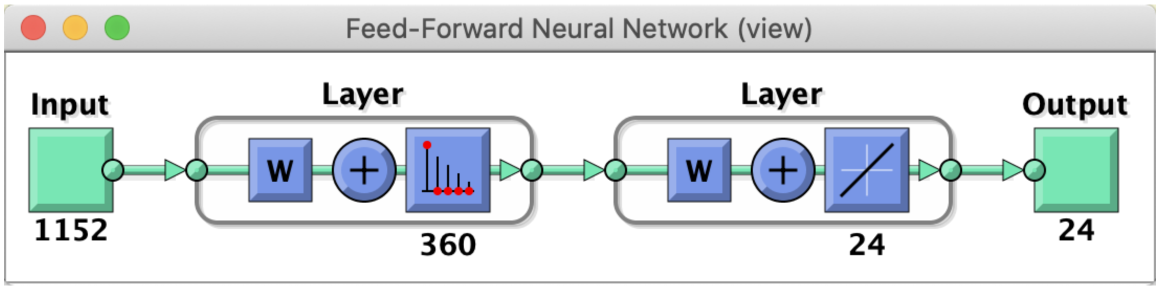


Figure 3: Structure of custom LVQ

Radial Basis Function (RBF) Networks with k-means

Radial basis function is structurally similar to the MLP, with input, hidden and output layer inside. Specifically, the RBF networks could only contain one hidden layer, with a non-linear transformation function, Gaussian Radial Function, to expands in the high dimensional space for classification. The following output layer applies a linear transformation from the hidden space to the output space (Figure 4).

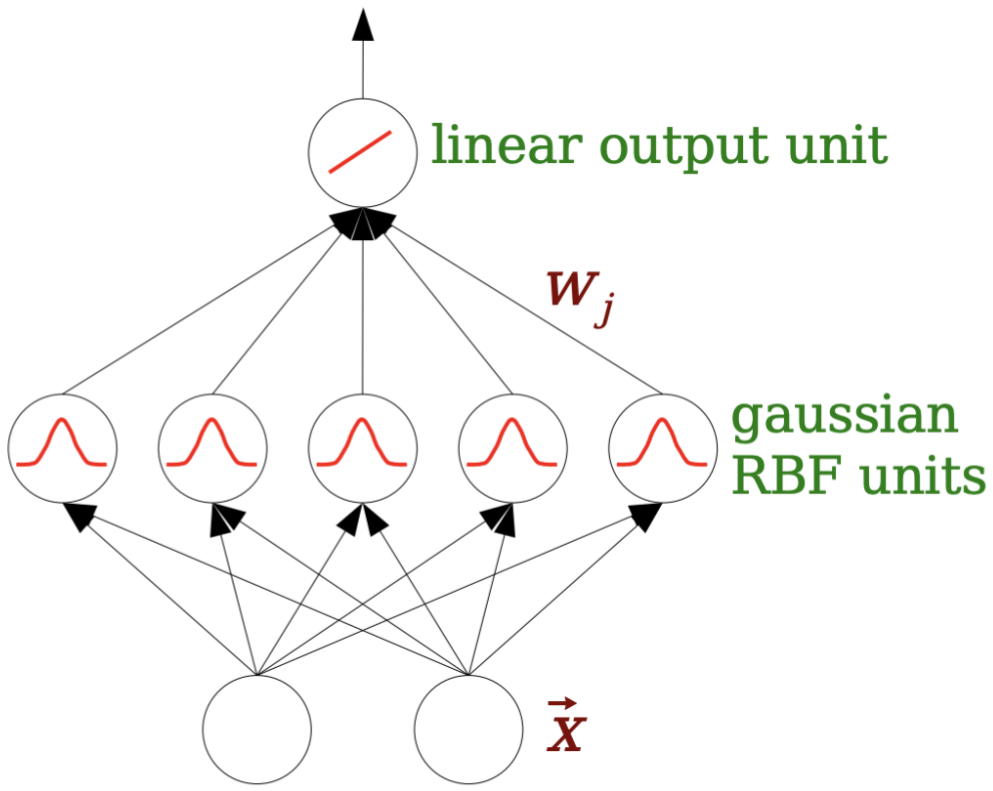


Figure 4: Structure of custom RBF network

$$Gaussian\ basis\ function: \phi(x_i) = exp\left(\frac{-(x - c_i)^2}{2{\sigma_i}^2}\right)$$

$$Radial\ basis\ functionnetwork: F(x) = \sum_{i=1}^{n} w_i\, \phi(x_i)$$

In this case, C stands for the center of the divided cluster region returned by unsupervised learning algorithms (k-means or SOM); σ stands for the variance and calculated as the mean distance between each unit and its closest neighbor. Regarding the weights w_i, it could be computed by pseudo-inverse method.

In this RBF network, k-means clustering algorithm was applied to determine the center of each cluster. Since one of the goals in this project is to compare the performance in each algorithm, I, therefore, uniform the number of clusters in unsupervised learning algorithms to 360. Meanwhile, Euclidean distance was considered as the measurement of similarities in the samples.

Radial Basis Function (RBF) Networks with Self Organizing Map (SOM)

A self-organizing map (SOM), a typical unsupervised learning algorithm, is designed to produce a low-dimensional (2D in this case), discretized representation of the input space of the training samples[4]. Similar to LVQ algorithm, SOM applies competitive learning to preserve the topological properties of the high dimensional space. The SOM algorithm contains three processes, competition, cooperation, and synaptic adaption.

In this project, the reduced two-dimensional space was set to 18*20, which means a total of 360 neurons in this network, following the number of defined clusters in previous unsupervised learning algorithms. Hyperparameters used in this SOM network could be accessed in Table 2. I used 10 training steps for the initial covering and the initial neighbor size set to 3. Additionally, the layer topology function set to Hexagonal and Euclidean distance was used to measure the similarity.

In this RBF network, I would like to use the cluster center returned by the weights of input layer in SOM. Once obtained the center's coordinate, the following step was similar to the previous RBF with k-means.

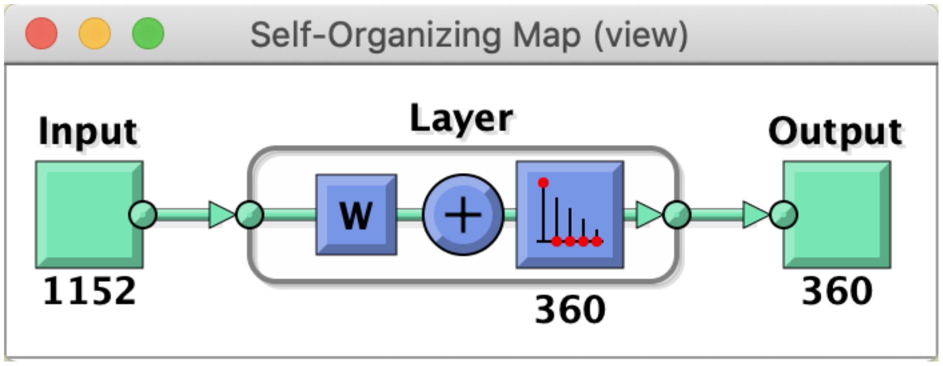


Figure 5: Structure of SOM network

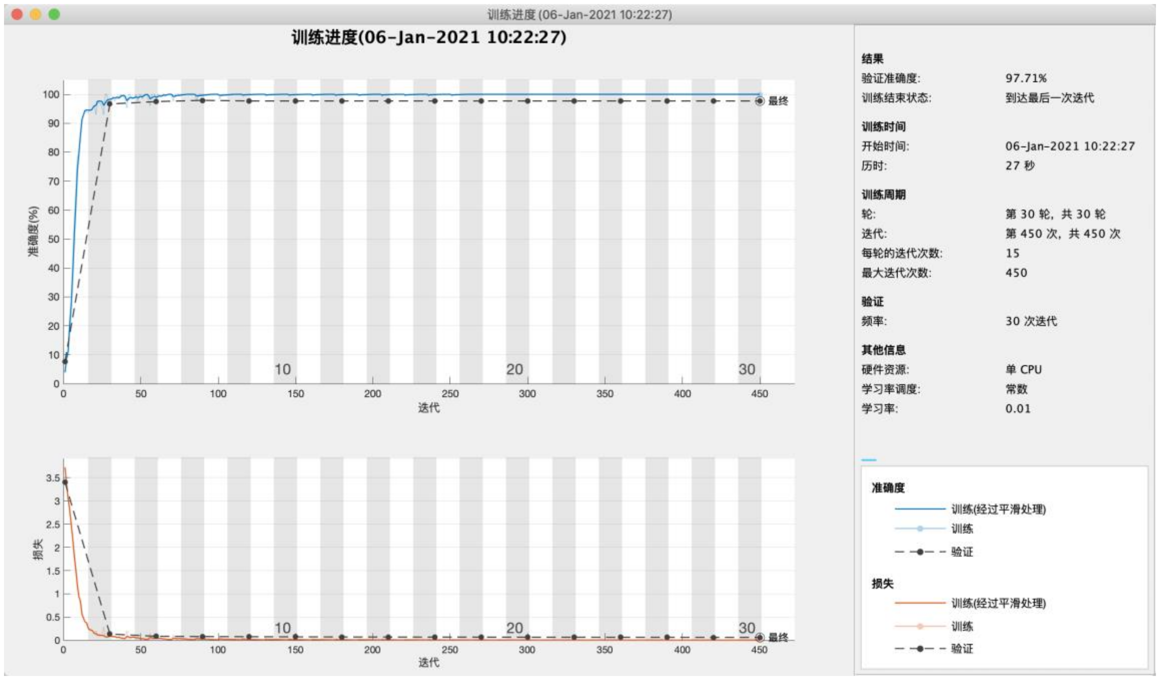
Cover step	Initial neighbor	Layer topology	Distance
10	80	Hexagonal	Euclidean

Table 2: Hyperparameter setting in SOM

Experimental results and analysis

Convolution Neural Network (CNN)

Figure 6 is the training process of CNN in this image classification problem. It is composed of two parts, the accuracy value trend indicated by blue line and the loss value trend indicated by red line. As can be seen from the plot that the model was robust and in high performance after 30 epochs, with accuracy about 97.8%. In this training process, stochastic gradient descent with momentum was used to update the weights and the learning rate was set to 0.01. For better visualization of the performance results, I applied a confusion matrix to demonstrate the final results (Figure 7).



结果	
验证准确度:	97.71%
训练结束状态:	到达最后一次迭代
训练时间	
开始时间:	06-Jan-2021 10:22:27
历时:	27 秒
训练周期	
轮:	第 30 轮，共 30 轮
迭代:	第 450 次，共 450 次
每轮的迭代次数:	15
最大迭代次数:	450
验证	
频率:	30 次迭代
其他信息	
硬件资源:	单 CPU
学习率调度:	常数
学习率:	0.01

Figure 6: Monitor of the convolutional neural network training process

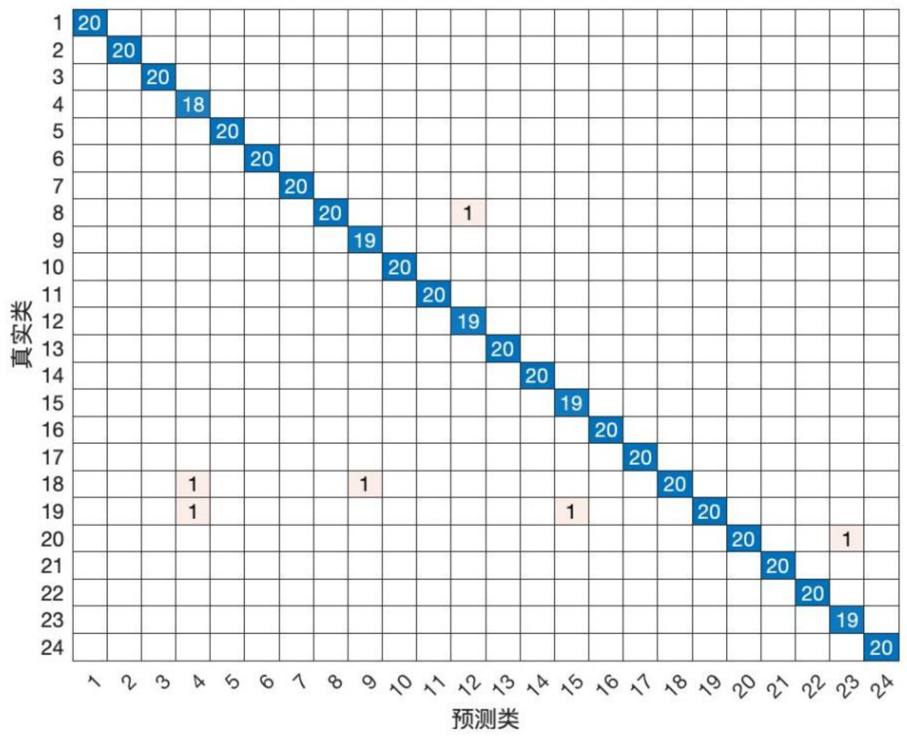


Figure 7: Confusion matrix of the convolutional neural network

Multi-layer Perceptron (MLP)

Mean squared error (MSE) was utilized to measure the performance of the MLP model. As can be seen from Figure 8, MSE reduced significantly before 100 epochs and started to be stable after that. The prediction model was devoid of overfitting or underfitting problems since all MSE values decreased to stable. The best validation performance is around 0.032 at epoch 1000. Additionally, the confusion matrix was plotted in Figure 9. Most labels could be successfully classified, but in some cases, deviation could occur.

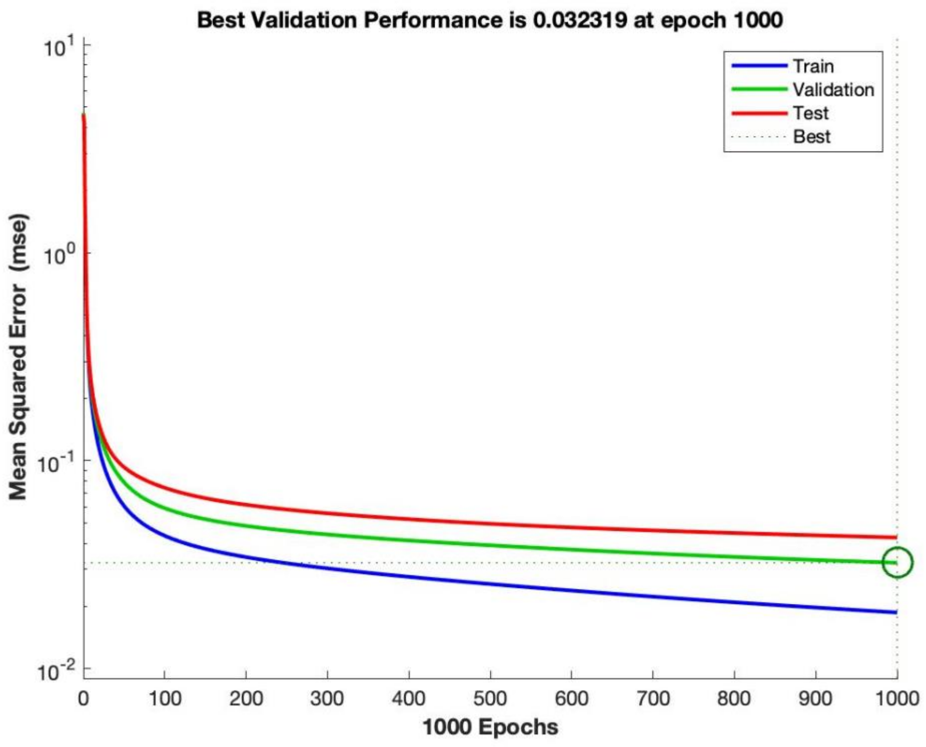
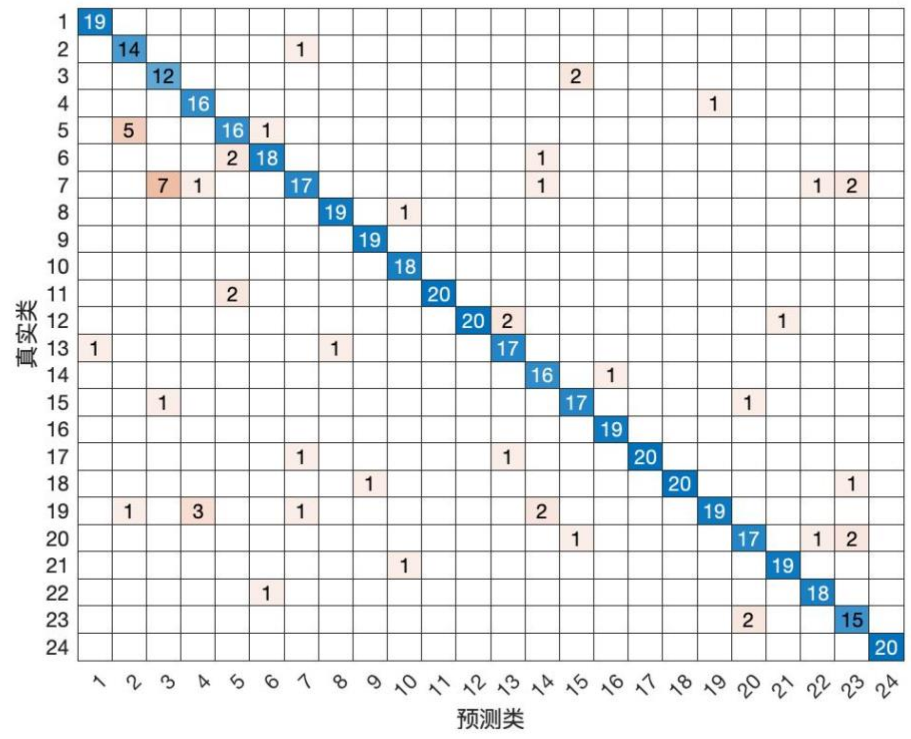


Figure 8: Mean squared error loss plot of MLP network



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	19																							
2		14					1																	
3			12												2									
4				16															1					
5		5			16	1																		
6					2	18								1										
7			7	1			17							1								1	2	
8								19		1														
9									19															
10										18														
11					2						20													
12												20	2								1			
13	1							1					17											
14														16		1								
15			1												17					1				
16																19								
17							1						1				20							
18									1									20					1	
19		1		3			1							2					19					
20															1					17		1	2	
21										1											19			
22						1																18		
23																				2			15	
24																								20

Figure 9: Confusion matrix of the multiple layer perceptron

Learning Vector quantization (LVQ)

Similarly, MSE was also considered as the measurement of LVQ network's performance. In this case, the best training performance could be reached at epoch 47, with a value of about 0.003. A confusion matrix was also made to evaluate the final prediction performance. Figure 11 indicates that, except for label 4 (“D” in character), the predictor could successfully classify the samples and obtained great results.

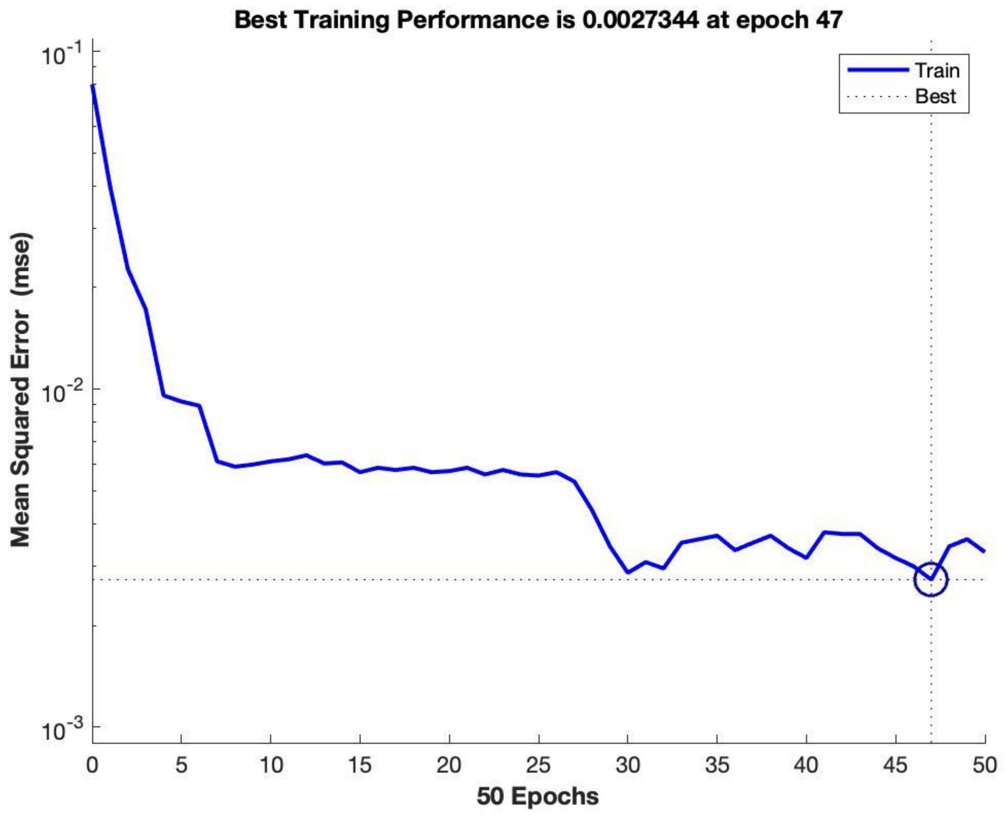


Figure 10: Mean squared error loss plot of LVQ network

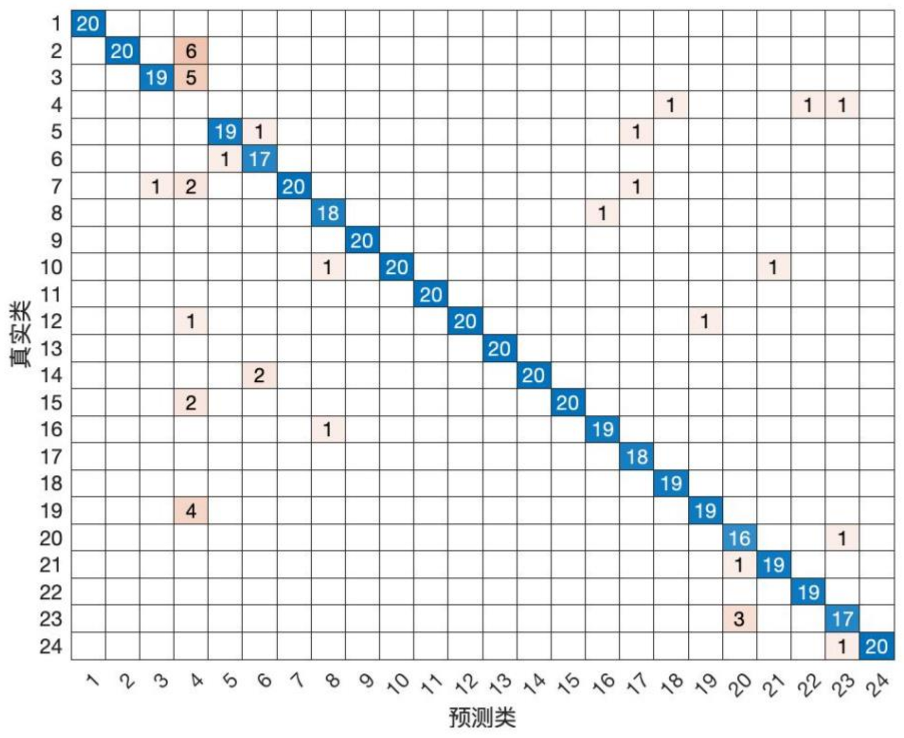


Figure 11: Confusion matrix of the LVQ network

Radial Basis Function (RBF) Networks

Figure 12 is the confusion matrix of RBF networks with k-means in determining the clusters' parameter. In this case, only three samples were misclassified, which could achieve high performance. In addition, since the accuracy was already high enough, I, therefore, did not include gradient descent to update the parameters.

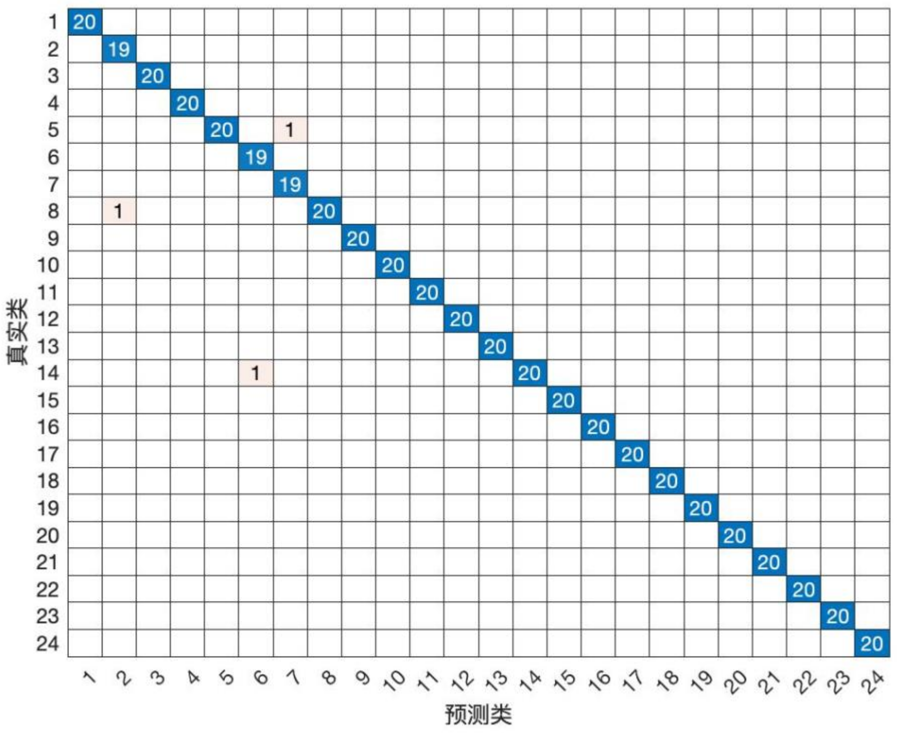


Figure 12: Confusion matrix of the RBF network with k-means

Additionally, SOM was also applied to determine the cluster center's position, necessary parameters in the RBF network. The topology of SOM, in this case, was 18*20 with hexagons represent the neurons.

In Figure 13, the blue hexagons represent the neurons with red lines indicate each neuron. The colors in the regions containing the red lines indicate the distances between neurons. The darker colors represent larger distances, and the lighter colors represent smaller distances.

The determined cluster centers were then fed into the RBF network for prediction. As can be seen from the confusion matrix, the RBF network with SOM in clustering could successfully divide the samples and make an accurate classification.

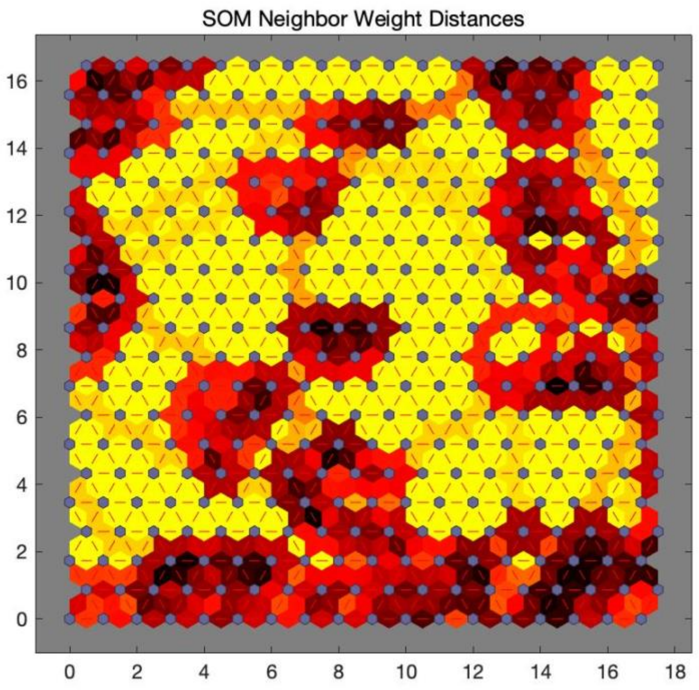


Figure 13: SOM Neighbor Weight Distances

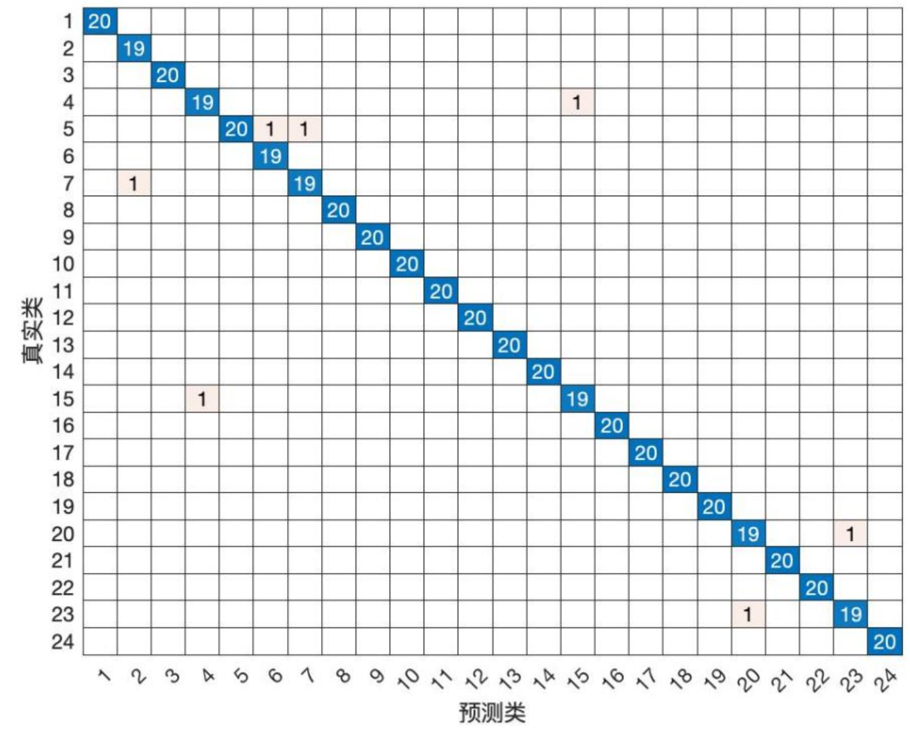


Figure 14: Confusion matrix of the RBF network with SOM

Comparison between different algorithms

Previously, I have utilized five algorithms to conduct classification problems. Therefore, it is important to return the most accurate model in this assignment. As can be seen from Table 3, the RBF network with k-means could achieve the highest performance, with both training and testing accuracy over 99%. Simultaneously, the lowest accuracy model could be the MLP, but still, the accuracy could be 88% in the testing set. Furthermore, the performance on training and testing samples did not deviate significantly, suggesting the absence of overfitting problems.

Accuracy (%)	MLP	CNN	LVQ	RBF-kmeans	RBF-SOM
Training	94.27	98.92	93.23	99.53	98.70
Testing	88.54	98.75	91.46	99.37	98.54
Rank	5	3	4	1	2

Table 3: Classification accuracy of different algorithms

Conclusion

To sum up, this project incorporated five algorithms to classify the segmented characters from the automobiles' license plates. The performance of each algorithm was evaluated by both training and testing accuracy as well as the visualized confusion matrix. Based on the results, the RBF network with k-means could achieve the highest performance, surpassing other competitors. However, the overall performance for RBF networks was already high enough, I, therefore, did not include the gradient descent to update the hyperparameters and this model may not maintain equal performance on the other datasets. In the future, a gradient descent algorithm should be included to update the hyperparameters in each epoch of the RBF network and achieve high and robust performance.

Appendix

You could download and check the code from this repository:
https://github.com/yuxuanwu17/INT301_Assessment2

Reference

[1] "What is a multi-layered perceptron?" https://www.educative.io/edpresso/what-is-a-multi-layered-perceptron (accessed Nov. 26, 2020).

[2] "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way | by Sumit Saha | Towards Data Science." https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (accessed Jan. 04, 2021).

[3] "Learning Vector Quantization for Machine Learning." https://machinelearningmastery.com/learning-vector-quantization-for-machine-learning/ (accessed Jan. 05, 2021).

[4] "Self Organizing Maps. Recently, I learned about SOMs while… | by Abhinav Ralhan | Medium." https://medium.com/@abhinavr8/self-organizing-maps-ff5853a118d4 (accessed Jan. 05, 2021).