Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2021, 8(7):81-89**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Handwritten digital recognition based on SVM, KNN, MLP, and Logistic Regression models

Xiaolin Wang, Yingcui Du, Shulong Dong, Chenchen Li, Yuan Fu

School of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo 255000, China

Abstract As the only universal symbol in the world, Arabic numerals play an irreplaceable role in all walks of life. With the development of science and technology, more and more data information needs to be input into the computer and then processed. Because of the low efficiency, the method of manual identification of numbers on paper is not suitable for the identification of massive data. As a branch of optical character recognition technology, the main research content of handwritten numeral recognition is how to use computer to automatically recognize Arabic numerals written on paper. At present, handwritten numeral recognition technology has more and more influence on the development of all walks of life. At the same time, due to the relatively simple implementation of handwritten numeral recognition, it can be used as a touchstone for other complex problems and algorithms. Although there are only 10 Arabic numerals in 0-9, and the strokes are relatively simple, due to the different handwriting habits of different countries and individuals, as well as the small differences between the numbers, the universality of handwritten numeral recognition is poor. To sum up, how to recognize hand written digits accurately, efficiently and with low rejection rate has always been one of the hot issues in the research field.

Keywords machine learning, handwritten numeric recognition, confusion matrix

Introduction

In recent years, scholars from all over the world have been studying digital recognition, and have made great achievements in handwritten digital recognition.

Cheng-Lin Liu [1], using 10 structural features such as contour features, self-structure features, curvature, and eight classifiers, testedon a test set of CENPARMI,CEDAR, and MNIST databases with a test recognition rate of 99.58 percent. But computing and storage are expensive. Loo-Nin Teow used biovision to build a handwritten digital recognition model, and the linearly separable features extracted from the model reduced the misperception rate to 0.59% on the MNIST training set. JinhaiCai [3] suggests a way to integrate statistical and structural information for unconstrained handwritten numerical recognition. This method improves the modeling of state time in conventional HMM by using the probability of state duration adaptive transfer, and uses macro states to overcome the difficulty of HMM modeling pattern structure, which has greatly improved speed and accuracy.

Huang Qiaoqiao's [4] handwritten digital recognition system based on BP neural network designed by Visal C.6.0 verifies the feasibility of applying BP neural network to handwritten digital recognition, and has a good recognition rate. Wang Qiang's [5] method of combining PCA (Principal Component Analysis, main component analysis) and CNN was conducted on the SVHN data set in an attempt to improve the recognition rate of characters in natural scenes. Cao Dan [6] has built and implemented a handwritten digital recognition model based on The Hopfield Neural Network, and the error rate and accuracy rate are more ideal than the RECOGNITION method of THE BP network.

German, Tsinghua University and Shanghai Jiaotong University [7], such as the handwriting digital recognition algorithm based on the characteristics of compressed word structure, selected a large number of samples to form different training sets and test sets, the use of advanced technology for high-volume testing, in accuracy and speed performance has been further improved, but there are no conditions for the handwriting numbers, its recognition effect still needs to be improved.

Handwritten digital recognition based on deep learning has high accuracy and low resource requirements, which provides theoretical reference for our opponent to write digital recognition application.

2 Data read

The handwritten digital recognition dataset consists of 11092 handwritten digital pictures with0-9 writing, each of which is a 28*28pixel“.bmp” format picture with black and white characters.

For this handwritten digital recognition experiment, a total of SVM, KNN, MLP, Logistic Regression, Ridge, Lasso and other 6 model opponents to write digital recognition experiments. Import the appropriate Python library based on the relevant model you choose.

Defines the function “get_file_list” that gets thefile in the “.bmp” format and gets the file underthe specified path.

Defines the function that resolves the name of the picture “file, get, img_name_str”, and resolves the name of the “.bmp” file.

3. Data pre-processing

Define the picture processor “img2vector”, convert all pictures into grayscale pictures of a size of 28*28, then normalize the grayscale values, and finally convert them to a matrix of 1 *784.

Defines the read function “read_and_convert”, traverses each picture, reads all the data, and then saves the handwritten numeric matrix and label values in data Mat and data Label, respectively.

Define “read_data” functions for reading and pre-processing of rival write digital datasets.

4. Predictive model

In the six models of SVM, KNN, MLP, logistic regression, ridge and Lasso, the scores of ridge and lasso are too low, so SVM, KNN, MLP and logistic regression models are selected to carry out the recognition experiment.

4.1. Support Vector Machine (SVM) model

4.1.1. Introduction to the Support Vector Machine (SVM) model

The support vector machine method is a classic supervised machine learning method that includes a variety of technical knowledge, such as maximum interval hyperplane, Mercer core, convex secondary planning and relaxation variables, and can be used to solve the problem of classification ii. In the case of linearly separable samples, the interval maximization method can be used to classify. The specific content is to assume that there are *n* samples in a given training sample set, i.e.:

$$D = \{(x_1, y_1), (x_2, y_2) \dots \dots (x_n, y_n)\}, \qquad y_i \in \{-1, +1\}$$

Where $x_i$the table shows the first sample consisting of m-dimensional features, the *ith* sample y_i is classified. The basic idea of SVM is to find a hyperplane in the sample space of known training samples, which $w^T x + b = 0$separates different samples while maximizing the distance from the support vector (i.e., the training sample closest to the hyperplane) to the hyperplane, which is referred to as the hyperplane with the "maximum interval". By looking for the hyperplane of the maximum interval, you find $y_i(w^T x + b) \geq 1 the\ w$ and $b$that satisfy the about bundle and maximize it. $\frac{2}{\|w\|}$

When the original sample is linearly non-divided, the sample feature space can be mapped to the higher dimensional sample feature space using the appropriate nuclear function, so that the sample is linearly divided within the higher dimensional feature space. The classification accuracy of the SVM is subject to the letter of approval number of effects, so choosing the right nucleus function is one of the important features of using SVM algorithms. Since SVM is a two-classification machine learning algorithm, that is, the classification results are either A or B, but in real life, many multi-classification scenarios, in order to solve the transition from two classifications to multi-classification, there are one-to-one (OneVSOne, OVO) and one-to-one (Onevs Rest, OVR) two common methods. The idea of solving the problem in the OVO method is that assuming that there are *n* categories of classification tasks, combining the *n* categories in two or two categories, will result in *n*(*n*-1)*2 two classification tasks. During the test, the new sample is submitted to all classifiers at the same time, resulting in *n*(*n*-1)*2 results, and the final test results combine all the results, using the most predicted categories as the result of the final classification. The OVR algorithm selects a sample of one class as a positive example at a time in each training, while the sample for all other classes trains *N* classifiers as negative examples. When testing, if only one classifier is predicted to be a positive class, the corresponding category is marked as the final classification result.

4.1.2. Support Vector Machine (SVM) model definition

```
def svm_model(X_train, y_train, X_test, y_test):
    """
    SVM
    """
clf = svm. SVC(C=1.0
               , kernel='rbf'
               , gamma='auto')
clf.fit(X_train, y_train)
    score = clf.score(X_test, y_test)
    result = clf.predict(X_test)
    print(f'SVMscore{score}')
    return result
```

4.2 K-Nearest Neighbor (KNN) model

4.2.1 Introduction to the K-Nearest Neighbor (KNN) model

The KNN algorithm results in categories for the sample to be classified by selecting the *K* sample values in the classification collection that are most similar to the sample to be classified. KNN algorithm belongs to classification and logistic regression algorithm, KNN only exists forward propagation process, there is no learning process, is a simple and effective classification algorithm in machine learning. Replace a category collection with a handwritten number collection for handwritten numeric recognition.

There are four core points in the whole process of KNN algorithm implementation: 1) the selection of classification collections, 2)the choice of *K* values, 3) vector distance measurement rules, and 4) decision scenarios. Among them, the selection of training data set and the choice of *K* value need comprehensive consideration, after the general assembly to reduce the efficiency of the whole algorithm implementation, too small will affect the accuracy of recognition; Vector distance measurement rules are used to determine the similarity of samples to be classified with those in the classification collection; Decision-making schemes are used to determine which category the samples to be classified belong to, and good decision-making schemes can greatly improve the accuracy of classification.

4.2.2 K-Nearest Neighbor (KNN) model definition

```
def knn(X_train, y_train, X_test, y_test):
    """
    KNN
    """
clf = neighbors. KNeighborsClassifier(n_neighbors=5)
clf.fit(X_train, y_train)
    score = clf.score(X_test, y_test)
    result = clf.predict(X_test)
    print(f'KNNscore{score}')
    return result
```

4.3. Multi-Layer Perceptron (MLP) model

4.3.1. Introduction to the Multi-Layer Perceptual (MLP) model

Multi layer perceptron neural networks (MLP) was invented by Frank Rosenblatt in the 1950s and 1960s. It accepts several binary inputs, x_1,x_2,..., and produces a binary output. In addition to the input / output layer, it can have multiple hidden layers in the middle. The simplest MLP only contains one hidden layer, which is a three-layer structure, as shown in Figure 1. MLP is a feed forward and supervised artificial neural network structure. Multi layer perceptron can contain multiple hidden layers to realize the classification modeling of nonlinear data. MLP divides the data into training set, test set and test set. The training set is used to fit the parameters of the network, the test set is used to prevent over training, and the test set is used to evaluate the effect of the network and applied to the total sample set. When the dependent variable is classified, the MLP neural network divides the records into the most suitable types according to the input data. Back propagation algorithm, which is often used by MLP for learning, is a standard supervised learning algorithm in the field of pattern recognition, and continues to be a research topic in the field of computational neurology and parallel distributed processing. MLP has been proved to be a general function approximation method, which can be used to fit complex functions or solve classification problems.

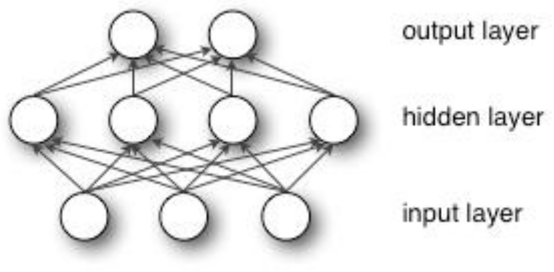

Figure 1: Multi-Layer Perceptual (MLP) model structure

4.3.2. Multi-Layer Perceptual (MLP) model definition

```
def mlp(X_train, y_train, X_test, y_test):
clf = MLPClassifier(hidden_layer_sizes=(50,),
max_iter=10,
                    alpha=1e-4,
                    solver='sgd',
                    verbose=10,
random_state=1,
learning_rate_init=.1
                    )
clf.fit(X_train, y_train)
   score = clf.score(X_test, y_test)
   result = clf.predict(X_test)
   print(f'MLPscore{score}')
   return result
```

4.4. Logistics Regression model

4.4.1. Introduction to the Logistics Regression model

Logistic regression algorithm is a classical classification method, which can predict the classification of samples and give the probability that samples belong to the classification. Its model is a decision function represented by the probability of a condition, i.e., the sample feature vector $P(y=1|x)=\frac{e^{w\cdot x}}{1+e^{w\cdot x}}x$ is the input data, the label is the output data, and $y\in\ \{0,1\}w$ is the weight vector is the parameter of the logistic regression model, which is the product $w\cdot x$of w and x.

The principle of the logistic regression algorithm is: first, the decision function is represented in the form of conditional probability, and then the best parameter w for the training set sample is obtained by maximizing the principle of the icing on the probability function, that is, the best model is obtained, and finally the new sample is analyzed and predicted by the model. The diagram of the logistic regression model is shown in the following image:

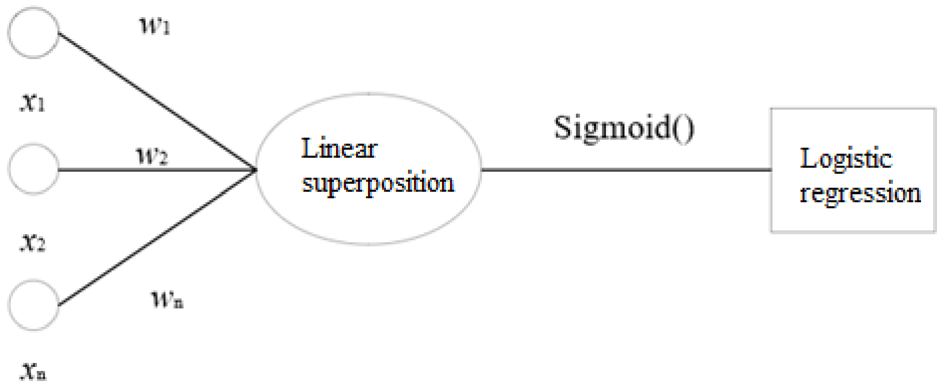

Figure 2: Logistics Regression model diagram

4.4.2. Logistics Regression model definition

```
def logisticregression(X_train, y_train, X_test, y_test):
    """
    Logistic
    """
clf = LogisticRegression(penalty='l2'
                         , solver='saga'
                         , multi_class='auto'
                         )
clf.fit(X_train, y_train)
    score = clf.score(X_test, y_test)
    result = clf.predict(X_test)
    print(f'Logisticscore{score}')
    return result
```

5. Model reviews

5.1. Confusion matrix

A confusion matrix is a situational analysis table that summarizes the prediction results of a classification model in data science, data analysis, and machine learning, which summarizes records in a data set in matrix form according to two criteria of classification judgment made by the real category and the classification model. Each column of the confusion matrix represents a prediction category, and the total number of columns represents the number of data predicted for that category, and each row represents the true attribution category of the data, and the total number of data for each row represents the number of instances of data for that category.

5.2. Confusion matrix Extends the various evaluation indicators

Accuracy, recall rate, and F-value are all important evaluation indicators for selecting goals.

(1) If an instance is a positive class, but is predicted to be a positive class, it is a true class (True Postive TP).

(2) True Negative TN if an instance is a negative class but is predicted to be negative(True Negative TN).

(3) False Postive FP if an instance is a negative class but is predicted to be a positive class(False Postive FP).

(4) If an instance is a positive class but is predicted to be negative, it is a false negative FN(False Negative FN).

TP stands for the correct number of matches; FP stands for false positives, no matches are correct; FN stands for false positives, no correct matches are found; and TN represents the correct number of non-matchings.

Accuracy (precision) is the prediction of positive classes as positive classes / all predictions as positive classes. That is: TP/(TP-FP).
Recall rate (recall)-Predicts a positive class as a positive class / All positive positive classes. That is: TP/(TP-FN).
F-value (f1-score)s accuracy s recall rates 2 / (accuracy rate s recall rate). (The F value is the reconciled average of accuracy and recall rate).
support:The number of times each label appeared.
The "classification_report" function in sklearn can be used to display a text report of the main classification metrics, showing indicator information for each class.

6. Model evaluation results

Output of recognition results, fitting accuracy results, and confusion matrices for handwritten numeric recognition of different models.

6.1. SVM model evaluation results

Table 1: SVM model recognition results

	precision	**recall**	**f1-score**	**support**
0	0.92	0.96	0.94	218
1	0.88	0.93	0.90	244
2	0.89	0.89	0.89	226
3	0.89	0.85	0.87	234
4	0.87	0.91	0.89	211
5	0.77	0.87	0.82	195
6	0.91	0.93	0.92	215
7	0.86	0.84	0.85	204
8	0.91	0.82	0.86	230
9	0.90	0.82	0.86	242

Table 2: SVM Model Fit Accuracy

	precision	**recall**	**f1-score**	**support**
Accuracy	-	-	0.88	2219
macro avg	0.88	0.88	0.88	2219
Weighted avg	0.88	0.88	0.88	2219

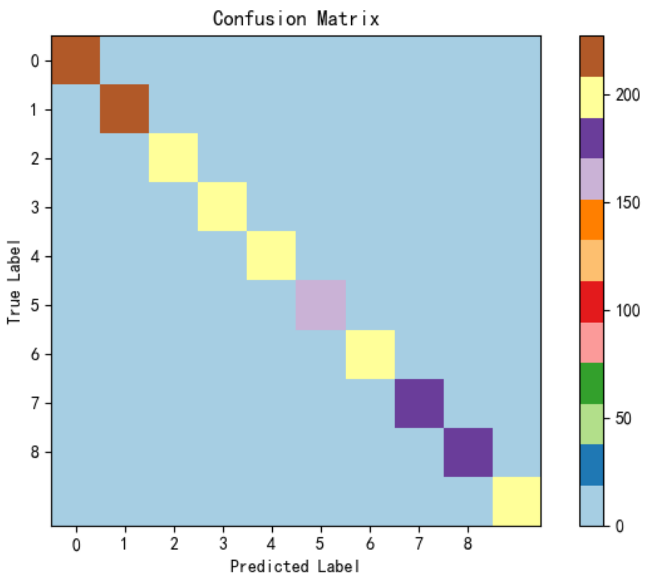

Figure 3: SVM model Confusion matrix

6.2. KNN model evaluation results

Table 3: KNN Model Recognition Results

	precision	**recall**	**f1-score**	**support**
0	0.91	0.99	0.95	218
1	0.81	0.99	0.89	244
2	0.94	0.88	0.91	226
3	0.92	0.94	0.93	234
4	0.95	0.89	0.92	211
5	0.95	0.90	0.92	195
6	0.98	0.96	0.97	215
7	0.85	0.88	0.87	204
8	1.00	0.82	0.90	230
9	0.92	0.90	0.91	242

Table 4: KNN Model Fit Accuracy

	precision	**recall**	**f1-score**	**support**
Accuracy	-	-	0.92	2219
macro avg	0.92	0.92	0.92	2219
Weighted avg	0.92	0.92	0.92	2219

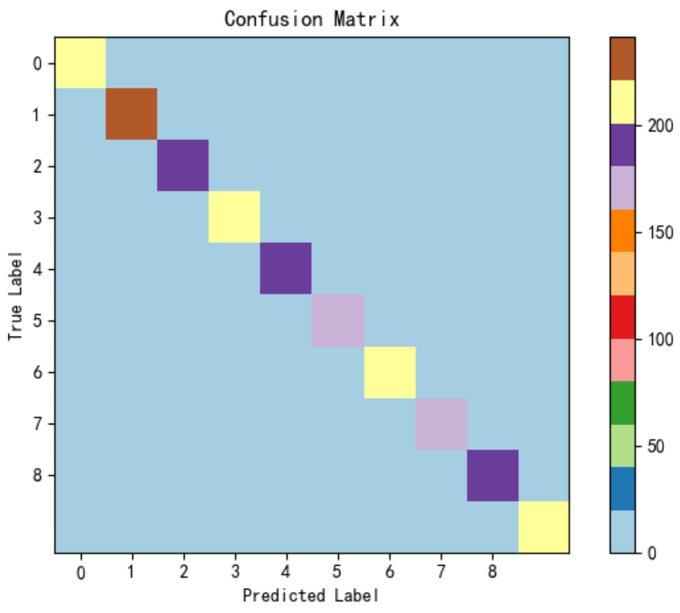

Figure 4: KNN model Confusion Matrix

6.3. Multi-layer perceptron (MLP) model evaluation results

Table 5: MLP model recognition results

	precision	**recall**	**f1-score**	**support**
0	0.95	0.97	0.96	218
1	0.94	0.96	0.95	244
2	0.95	0.92	0.93	226
3	0.91	0.90	0.91	234
4	0.88	0.94	0.91	211
5	0.86	0.92	0.89	195
6	0.94	0.95	0.95	215
7	0.93	0.88	0.90	204
8	0.92	0.89	0.90	230
9	0.93	0.89	0.91	242

Table 6: MLP Model Fit Accuracy

	precision	recall	f1-score	support
Accuracy	-	-	0.92	2219
macro avg	0.92	0.92	0.92	2219
Weighted avg	0.92	0.92	0.92	2219

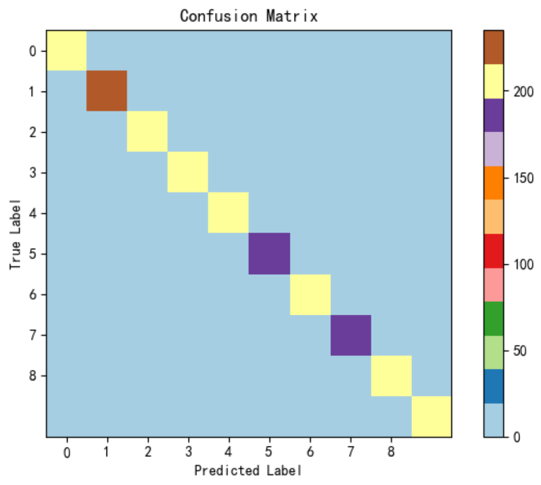

Figure 5: MLP model Confusion Matrix

6.4. Logistics Regression model evaluation results

Table 7: Logistic model recognition results

	precision	recall	f1-score	support
0	0.92	0.94	0.93	218
1	0.90	0.92	0.91	244
2	0.84	0.84	0.84	226
3	0.86	0.81	0.83	234
4	0.87	0.87	0.87	211
5	0.78	0.86	0.82	195
6	0.89	0.92	0.91	215
7	0.84	0.82	0.83	204
8	0.86	0.80	0.83	230
9	0.82	0.81	0.82	242

Table 8: Logistic model fit precision

	precision	recall	f1-score	support
Accuracy	-	-	0.86	2219
macro avg	0.86	0.86	0.86	2219
Weighted avg	0.86	0.86	0.86	2219

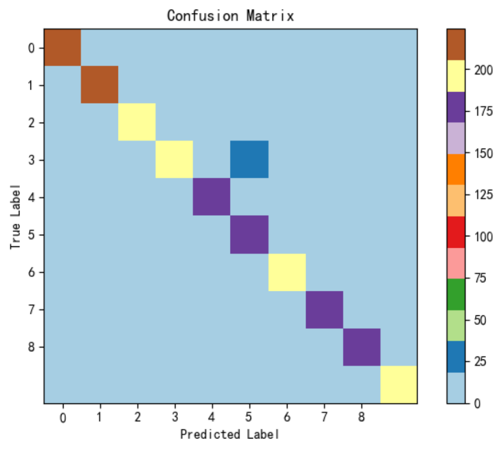

Figure 6: Logistic model Confusion Matrix

7. Summary

Extracting the F-values in it for evaluation is shown below:

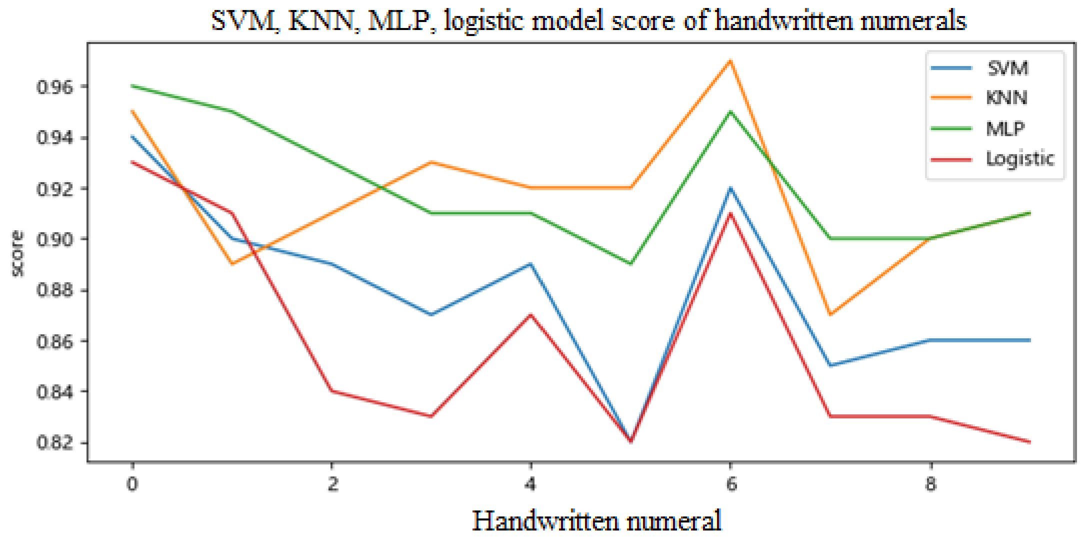

Figure 7: Each model predicts accuracy

Of the four models selected, the SVM model, the KNN model, the MLP model, and the Logistics Regression model scored 0.8806, 0.9099, 0.9225, and 0.8580, where the MLP model scored highest and the Logistics Regression model scored the lowest.

References

[1]. Liu C L, Nakashima K, Sako H, et al. Handwritten digit recognition: benchmarking of state-of-the-art techniques[J]. Pattern Recognition, 2003, 36(10):2271-2285.

[2]. Loo N T, Loe K F. Robust vision-based features and classification schemes for on-line handwritten digit recognition. Pattern Recognition, 2002,3 5:2355-2364.

[3]. Cai J, LiuZ Q. Integration of Structural and Statistical Information for Unconstrained Handwritten Numeral Recognition[J]. Pattern Analysis & Machine Intelligence IEEE Transactions on, 1999, 21(3):263-270.

[4]. Huang Qiaoqiao. Research on handwritten digital recognition system based on BP neural network. Central China Normal University, 2009.

[5]. Wang Qiang. Based on CNN's character recognition methods. Tianjin Normal University, 2014.

[6]. Cao Dan. Offline handwritten digital recognition based on Hopfield neural networks. Central South University, 2009.

[7]. Chen Zeus, Hu Wenxin. Simplifying speech synthesis of LSTM. Computer Engineering and Applications, 2018, 54 (003): 131-135.