Character Recognition: Classification and improvement with Hidden Markov Models

Qianying Lin

1. Introduction

Character recognition has great potential for improving human-machine interaction. The technology gains the recent improvements in classification methods. I test the Naïve Bayes, Multilayer Perceptron (MLP), and Linear Support Vector Machines (SVM) on a handwriting word dataset. Based on the results, SVM achieves the best performance. I also show that Hidden Markov Models (HMMs) has the potential to improve the accuracy. Therefore, a combination of SVM and HMM might get the best accuracy on our target dataset.

2. Problem Definition and Algorithm

2.1 Task definition

Optical character reorganization is computerizing and interpreting both human and computer generated inputs in form of scanned images [1, 2]. This technique relieves us from painful job of template selection and tuning, and the recognition accuracies get improved significantly. It has applications in many areas, such as reading aid for blind, bank checks and conversion of any hand written document into structural text form [2].

In the early stage of optical character reorganization development, recognition techniques were based on templates or prototypes that were designed artificially, selected or averaged from few samples [2]. As the number of samples increased, this simple design methodology became insufficient to accommodate the shape variability of samples. Attention has been turned to classification methods learning from examples strategy, including statistical methods, artificial neural networks, support vector machines, and multiple classifier combination.

Hidden Markov Models were initially developed in the 1960's, and was applied to the problem of speech recognition since the 1970's. It increasingly superior results of speech recognition dramatically increased interest in applying HMMs to other difficult pattern recognition problems such as handwriting recognition.

In this paper, I discuss the results of Naïve Bayes, Multilayer perceptron, and linear support vector machine applied on the dataset. I also applied the HMMs methods on the above classification methods and discuss the performance on the dataset.

2.2 Classification Algorithm

I will give a brief overview of classification methods including statistical methods, artificial neural networks, and support vector machines.

2.2.1 Statistical Methods

Statistical classifiers based on the Bayes decision rule can be divided into parametric ones and nonparametric ones [2]. Parametric classifiers include the linear discriminant function, the quadratic discriminant function, the Gaussian mixture classifier and so on. Non-parametric methods, such as the nearest neighbor and KNN rules, the decision-tree, the subspace method, are not much used.

2.2.2 Artificial Neural Networks

Feed-forward neural networks, including multilayer perceptron, radial basis function network, the probabilistic neural network, higher-order neural network, have been widely applied to pattern recognition [2]. The connecting weights are usually adjusted to minimize the squared error on training samples in supervised learning. Using a modular network for each class was shown to improve the classification accuracy. S.Arora [1] proposed a MLP designed on some statistical features for handwritten characters recognition.

2.2.3 Support Vector Machines

Kernel based methods, including support vector machines primarily, kernel principal component analysis, and kernel Fisher discriminant analysis, are receiving increasing attention and have shown superior performance in pattern recognition [2]. An SVM is a binary classifier with discriminant function being the weighted combination of kernel functions over all training samples. Dong et al. used a one against-others scheme for large set Chinese character recognition with fast training [4]. They used a coarse classifier for acceleration but the large storage of SVs was not avoided.

2.3 Hidden Markov Model

Hidden Markov models are statistical models in which system being modeled is considered as a Markov process that has unobserved or hidden states [5, 6, 7]. In hidden Markov models, the state is not directly visible, but only the sequence of observations influenced by the state at a particular instance of time are visible and hence the name hidden Markov Models.

2.4 Dataset

OCR dataset contains handwritten words dataset collected by Rob Kassel at MIT Spoken Language Systems Group. It is a "clean" subset of the words and rasterized and normalized the images of each letter. Since the first letter of each word was capitalized and the rest were lowercase, the first letter was removed and only used

the lowercase letters.

3. Experimental Evaluation

3.1 Methodology

I used software WEKA to perform the Naïve Bayes, MLP, and SVM classification methods. Firstly, I removed the irrelevant data from the original file and extract them to a new file letter mod 1. csv. In the new file, the first column is the letter and the rest columns are the pixels. Each pixel value is 0 (blank) or 1 (words). Then I converted (cvs2arff.java) the new file (letter mod 1. csv) to an formatted arff file required by WEKA. I performed (project.java) the Naïve Bayes and Multi-layer perceptron classification by referencing the WEKA library, and used WEKA to perform LibSVM classification.

After experiments, I outputted the summary results of the three classification methods, which are summarized the section 4. I also outputted all predicted results of Naïve Bayes (The other two classification results do not follow the sequence) and used it as the input for the next HMMs improvement.

For the HMMs experiment, I first prepared the required data with characterReco.java, and then used the statistical tools in MATLAB to perform HMMs (hmm.m). The following describes the way I build the model.

After editing the results from WEKA output, I built the transition table and emission table required for the HMMs on MATLAB. First of all, I extracted all the distinct characters that appear in the dataset and made a list of them. This list is functioned as the header for both the transition table and the emission table. Here we called it "header file".

To build the transition table, I firstly built a temporary table where each value represents the occurrences of different pairs of characters. The rows represent the previous character and the columns represent the next character. We count all pairs of two continuous characters for every word. For example, given a word "YES", I counted the occurrences of " Y-E " and "E-S" separately in the transition table. Then I normalized all values in this temporary table to build the real transition table (transi.csv). The normalization is to make sure that the sum of all values in each row equals to 1. In our word "YES" example, normalization means that the total probability of all characters following "Y" is 1. In order to avoid the problem caused by nonexistence (count = 0), I used Laplace smoothing where I added one to each occurrence and added number of character to the total occurrence. Therefore, each probability in the transition table represents that the possibility of column character appear after the row character.

For emission table, I also began with a temporary table where each value represents the occurrences of pair of characters from the formatted Naïve Bayes prediction output from WEKA. The rows represent the original characters and the columns represent the predicted characters. For example, if the original character is "Y" and the predicted value is "E", we will count this as a "Y (row) – E (column)". Similarly, I used Laplace smoothing normalization to build the final emission table (emiss.csv). So each value in the table represents the probability for row character being predicted to the column character.

To build the sequence list required by HMMs model in the MATLAB, I converted the sequence of characters into a list of indexes referencing to the "header file", and output the sequence as two files (oriniSeq_mod.csv and preSeq_mod.csv). For example, given a word "NO", if the index of character "N" is 3 and index of "O" is 5 in the "header file", we will convert the word "NO" to "3, 5" in the output file.

With original sequence file, Naïve Bayes prediction sequence file, transition table and emission table, I perform the re-classification using the HMMs toolkit in MATLAB. For each predicted word results. I used the HMMs to estimate the new sequences of the Naïve Bayes sequence based on the transition table and the emission table. Then I compared the new sequences with the original sequence, and counted the right predicted pairs. In this way, we calculated the accuracy after using HMMs method.

3.2 Results

We used the accuracy of classification to compare the performance of each method on this dataset. For each test, we use 10-fold cross-validation method.

Method	Parameter (if applicable)		Accuracy (%)
Naïve Bayes			61.5
Multiple Layer Perceptron	Training times	Layer number	
	10	1	15.6
	10	5	40.8
	10	1,5	18.5
	10	5,5	57.9
	10	10	58.8
	10	10,10	68.6
	30	10,10	69.8
	10	10,10,10	66.4
SVM			82.6
HMMs-Naïve Bayes			64.7

3.3 Discussion

The above experiment results show Naïve Bayes classification, as a simple but effective method, works fine on the dataset. The accuracy of MLP classification varies with the training times and layer numbers. With low number of layers and units, MLP performs extremely poorly. But with layer larger than 1 and/or units larger than 5, the prediction accuracy increases significantly. However, increasing the number of layers does not necessary improves the accuracy. The results with layer of ”10, 10, 10” has a lower accuracy than that with layer of “10, 10”. So the improvement of accuracy of MLP by increasing number of layer and units is limited. Moreover, MLP is very slow compared with Naïve Bayes. The accuracy with LibSVM is much better than the previous two methods, reaching as high as 82.6%.

Comparing the accuracies of Naïve Bayes and HMMS-Naïve Bayes, the results also show that HMMs improves the accuracy by 3.2%. I didn't test the result from other classification (because of output sequence from WEKA is not consistent with the input sequence), so I could not test whether HMMs will improve accuracies for sure. However, because HMMs improvement is independent from the previous classification method, it is probable that HMMs will improve the accuracy for other classifications as it did for Naïve Bayes. Therefore, I think the best way is to combine HMMs and SVM.

4. Future Work

Despite the improvements, the recognition accuracies of either machine-printed characters on degraded document image or freely handwritten characters are still insufficient. My work shows the effectiveness in SVM and HMM. So the future will try to combine these two methods together. Moreover, I need to test the classification on other dataset, such as image datasets with larger noise, to make sure that this classification works well in other datasets.

5 Conclusions

1 Linear SVM performs better than Naïve Bayes and MLP classifications. For MLP classification, If you increase the training time or layer numbers, you can also improve the results.
2 HMMs has the potential to improve the classification accuracy.

6. Bilbiography

[1] Arora, Sandhya, Debotosh Bhattacharjee, Mita Nasipuri, Latesh Malik, Mohantapash Kundu, and Dipak Kumar Basu. "Performance comparison of SVM and ANN for handwritten devnagari character recognition." arXiv preprint arXiv:1006.5902 (2010).

[2] Kumar, Parveen, Nitin Sharma, and Arun Rana. "Handwritten Character Recognition using Different Kernel based SVM Classifier and MLP Neural Network

(A COMPARISON)." International Journal of Computer Applications53, no. 11 (2012).

[3] Rabiner, Lawrence R. "A tutorial on hidden Markov models and selected applications in speech recognition." Proceedings of the IEEE 77, no. 2 (1989): 257-286.

[4] Dong, Jian-xiong, Adam Krzyżak, and Ching Y. Suen. "Fast SVM training algorithm with decomposition on very large data sets." Pattern Analysis and Machine Intelligence, IEEE Transactions on 27, no. 4 (2005): 603-618.

[5] Rabiner, Lawrence R., Jay G. Wilpon, and Frank K. Soong. "High performance connected digit recognition using hidden Markov models."Acoustics, Speech and Signal Processing, IEEE Transactions on 37, no. 8 (1989): 1214-1225.

[6] Rashid, Sheikh Faisal. "Optical Character Recognition-A Combined ANN/HMM Approach." PhD diss., TU Kaiserslautern, 2014.

[7] Shu, Han. "On-line handwriting recognition using hidden Markov models." PhD diss., Massachusetts Institute of Technology, 1996.

7. Original Result Outputs

Cross-Validation, Folds =10

1 Naïve Bayes

Correctly Classified Instances	32031	61.4185 %
Incorrectly Classified Instances	20121	38.5815 %
Kappa statistic	0.5926	
Mean absolute error	0.0303	
Root mean squared error	0.1603	
Relative absolute error	41.809 %	
Root relative squared error	84.2432 %	
Coverage of cases (0.95 level)	71.1574 %	
Mean rel. region size (0.95 level)	5.4357 %	
Total Number of Instances	52152	

2 MultilayerPerceptron

unchanged parameter: learningRate 0.3, momument 0.2

1) HiddenLayer =1, training time = 10

Correctly Classified Instances	8092	15.5162 %

Incorrectly Classified Instances 44060 84.4838 %
Kappa statistic 0.0678
Mean absolute error 0.0693
Root mean squared error 0.1856
Relative absolute error 95.7559 %
Root relative squared error 97.5361 %
Coverage of cases (0.95 level) 95.6799 %
Mean rel. region size (0.95 level) 79.7989 %
Total Number of Instances 52152

2) HiddenLayer =5, training time = 10

Correctly Classified Instances 21302 40.846 %
Incorrectly Classified Instances 30850 59.154 %
Kappa statistic 0.3583
Mean absolute error 0.0559
Root mean squared error 0.1666
Relative absolute error 77.2084 %
Root relative squared error 87.561 %
Coverage of cases (0.95 level) 92.1345 %
Mean rel. region size (0.95 level) 54.2494 %
Total Number of Instances 52152

3) HiddenLayer =1, 5, training time = 10

Correctly Classified Instances 9649 18.5017 %
Incorrectly Classified Instances 42503 81.4983 %
Kappa statistic 0.1027
Mean absolute error 0.0675
Root mean squared error 0.1838
Relative absolute error 93.1856 %
Root relative squared error 96.598 %
Coverage of cases (0.95 level) 96.2226 %
Mean rel. region size (0.95 level) 74.9719 %
Total Number of Instances 52152

4) HiddenLayer =5, 5, training time = 10

Correctly Classified Instances 30229 57.9633 %
Incorrectly Classified Instances 21923 42.0367 %
Kappa statistic 0.5521
Mean absolute error 0.0439
Root mean squared error 0.1498
Relative absolute error 60.6502 %
Root relative squared error 78.7611 %
Coverage of cases (0.95 level) 88.0887 %

Mean rel. region size (0.95 level) 29.9372 %
Total Number of Instances 52152

5) HiddenLayer =10, training time = 10

Correctly Classified Instances 30660 58.7897 %
Incorrectly Classified Instances 21492 41.2103 %
Kappa statistic 0.5614
Mean absolute error 0.0406
Root mean squared error 0.1486
Relative absolute error 56.0928 %
Root relative squared error 78.129 %
Coverage of cases (0.95 level) 85.9162 %
Mean rel. region size (0.95 level) 22.5472 %
Total Number of Instances 52152

6) HiddenLayer = 10, 10, training time = 10

Correctly Classified Instances 35797 68.6397 %
Incorrectly Classified Instances 16355 31.3603 %
Kappa statistic 0.6663
Mean absolute error 0.0331
Root mean squared error 0.1334
Relative absolute error 45.7362 %
Root relative squared error 70.0984 %
Coverage of cases (0.95 level) 88.2919 %
Mean rel. region size (0.95 level) 16.9857 %
Total Number of Instances 52152

7) HiddenLayer = 10, 10, training time = 30

Correctly Classified Instances 36438 69.8688 %
Incorrectly Classified Instances 15714 30.1312 %
Kappa statistic 0.6795
Mean absolute error 0.0306
Root mean squared error 0.1319
Relative absolute error 42.3286 %
Root relative squared error 69.3299 %
Coverage of cases (0.95 level) 86.8692 %
Mean rel. region size (0.95 level) 13.027 %
Total Number of Instances 52152

8) HiddenLayer = 10, 10, 10, training time = 10

Correctly Classified Instances 34611 66.3656 %
Incorrectly Classified Instances 17541 33.6344 %

Kappa statistic	0.642
Mean absolute error	0.0353
Root mean squared error	0.1365
Relative absolute error	48.7595 %
Root relative squared error	71.7318 %
Coverage of cases (0.95 level)	89.6054 %
Mean rel. region size (0.95 level)	21.57 %
Total Number of Instances	52152

3 SVM

(1) batch size = 1, type = c-svc, degree = 1, kernel type= linear, fold =2

Correctly Classified Instances	42378	81.2586 %
Incorrectly Classified Instances	9774	18.7414 %
Kappa statistic	0.8007	
Mean absolute error	0.0144	
Root mean squared error	0.1201	
Relative absolute error	19.9152 %	
Root relative squared error	63.1121 %	
Coverage of cases (0.95 level)	81.2586 %	
Mean rel. region size (0.95 level)	3.8462 %	
Total Number of Instances	52152	

(2) batch size = 1, type = c-svc, degree = 1, kernel type= linear, fold =10

Correctly Classified Instances	43104	82.6507 %
Incorrectly Classified Instances	9048	17.3493 %
Kappa statistic	0.8155	
Mean absolute error	0.0133	
Root mean squared error	0.1155	
Relative absolute error	18.4361 %	
Root relative squared error	60.7229 %	
Coverage of cases (0.95 level)	82.6507 %	
Mean rel. region size (0.95 level)	3.8462 %	
Total Number of Instances	52152	

4 HMMs Result

Naïve Bayes + HMMs
right predict number:
32145

right hmm predict number:
33753

total predict number:
52152

original accuracy:
61.6371

hmm accuracy:
64.7204