Combination of three classifiers with different architectures for handwritten word recognition

Simon Günter and Horst Bunke
Department of Computer Science, University of Bern
Neubrückstrasse 10, CH-3012 Bern, Switzerland
E-mail:{sguenter,bunke@iam.unibe.ch}

Abstract

The study of multiple classifier systems has become an area of intensive research in pattern recognition recently. Also in handwriting recognition, systems combining several classifiers have been investigated. In this paper the combination of three classifiers for handwritten word recognition with different architectures is studied. In addition a new ensemble method working with several base classifiers is applied and the results of the ensemble method are compared to the results of the combination of the three classifiers. In the experiments a large scale handwritten word recognition task is considered.

Keywords: handwritten word recognition, classifier combination, ensemble method

1 Introduction

The field of off-line handwriting recognition has been a topic of intensive research for many years. First only the recognition of isolated handwritten characters was investigated [29], but later whole words [27] were addressed. Most of the systems reported in the literature until today consider constrained recognition problems based on vocabularies from specific domains, e.g. the recognition of handwritten check amounts [14] or postal addresses [15]. Free handwriting recognition, without domain specific constraints and large vocabularies, was addressed only recently in a few papers [16, 23]. The recognition rate of such systems is still low, and there is a need to improve it.

The combination of multiple classifiers was shown to be suitable for improving the recognition performance in difficult classification problems [20, 33]. Also in handwriting recognition, classifier combination has been applied. Examples are given in [2, 21, 34]. Recently new ensemble creation methods, called ensemble methods, have been proposed in the field of machine learning, which generate an ensemble of classifiers from a single classifier [5]. Given a single classifier, the base classifier, a set of classifiers can be generated by changing the training set [3], the input features [13], the input data by injecting randomness [4], or the parameters and the architecture of the classifier [25]. Another possibility is to change the classification task from a multi-class to many two-class problems [6]. Examples of widely used methods that change the training set are Bagging [3] and AdaBoost [8]. Random subspace method [13] is a well-known approach based on changing the input features. A summary of ensemble methods is provided in [5].

Although the popularity of multiple classifier systems in handwritten recognition has grown significantly, not much work on the use of ensemble methods has been reported in the literature. An exception is [24], but this paper addresses only the classification of isolated characters, while the focus of the present paper is on the recognition of cursive words.

The contribution of the paper is two-fold. Firstly, three classifiers for handwritten word recognition with different architectures are introduced and results of their combination are presented. Secondly, the ensemble method presented in [9] is applied using the three classifiers. Unlike other ensemble methods, where a classifier is generated out of a single base classifier, this method use several base classifiers. The results of the combination and the results of the ensemble method are compared to each other.

Section 2 contains the description of the three handwritten word classifiers. Some methods to combine a set of classifiers are presented in Section 3. In Section 4 the ensemble method introduced in [9] is described in some detail. The results of the experiments

Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)
0-7695-2187-8/04 $20.00 © 2004 IEEE

are then given in Section 5 and, finally, conclusions are drawn in Section 6.

2 Handwritten word classifiers

In the experiments of this paper three base classifiers, C_1, C_2, and C_3, are used. All three classifiers use hidden Markov models (HMMs) and are similar to each other as described in the following. We assume that each handwritten word input to a classifier has been normalized with respect to slant, skew, baseline location and height (for details of the normalization procedures see [23]). A sliding window is moved from left to right over the word and at each position a feature vector is extracted. For each uppercase and lowercase character, a hidden Markov mode (HMM) [26] is build. For all HMMs the linear topology is used, i.e. there are only two transitions per state, one to itself and one to the next state. The character models are concatenated to word models. There is exactly one model for each word from the underlying dictionary. This approach makes it possible to share training data across different words. That is, each word in the training set containing character x contributes to the training of the model of x. Thus the words in the training set are more intensively utilized than in the case where an individual model is build for each word as a whole, and characters are not shared across different models.

The implementation of all systems is based on the Hidden Markov Model Toolkit (HTK), which was originally developed for speech recognition [35]. This software tool employs the Baum-Welch algorithm for training and the Viterbi algorithm for recognition [26]. The output of an HMM classifier is the word with the highest rank among all word models together with its score value.

The first classifier, C_1, is similar to the one described in [23]. Here the width of the sliding window is one pixel and nine geometric features are extracted at each position of the window. Thus an input word is converted into a sequence of feature vectors in a 9-dimensional feature space. The geometric features used in the system include the fraction of black pixels in the window, the center of gravity, and the second order moment. These features characterize the window from the global point of view. The other features give additional information. They represent the position of the upper- and lowermost pixel, the contour direction at the position of the upper- and lowermost pixel[1], the number of black-to-white transitions in the window, and the fraction of black pixels between the upper- and lowermost black pixel. In [23] a more detailed description of the feature extraction procedures can be found. The feature distributions in each state of an HMM are modeled by Gaussian mixtures. The training method of the classifier, which involves the determination of the number of Gaussians in each state and the number of training iterations, was optimized on a validation set, using a strategy described in [10]. In addition the number of states in each HMM is optimized by the Quantile method introduced in [36].

The second classifier, C_2, is a modified version of the classifier presented in [30, 31]. It uses a sliding window for feature extraction were the window width is 16 (the shift of the window between the extraction of a feature vector is still 1). The window is partitioned into 16 cells arranged in a 4×4 grid. The average grey value of the pixels of each cell is used as a feature. A Karhunen-Loeve transformation [17] is then applied to the feature vectors and only the first 14 components of the transformed feature vectors are used. Similarly to C_1, classifier C_2 also uses a training method optimized by a strategy presented in [10], and models the distribution of the features by Gaussian mixtures. In addition the number of states in each HMM is optimized by the Quantile method introduced in [36].

The third classifier, C_3, is a discretized version of classifier C_1. All feature vectors are grouped in a number, cn, of clusters and each feature vector is tagged with the cluster number it belongs to. After the clustering, only the tags of the feature vectors are considered. The probability of observing a feature vector corresponding to any of the cn tags must be set in the training phase for each state of the HMM, i.e. there are cn free parameters per state. The disadvantage of this approach is that we loose information, because we don't distinguish between the feature vectors of the same cluster. The main advantage is that, unlike in continuous HMM classifiers, we are getting probabilities instead of likelihood values as the output. For the experiments of this paper 110 clusters were used, i.e. $cn = 110$. Four iterations of the Baum-Welch algorithm are used in the training and the number of states per HMM was fixed to 14.

3 Combination methods

There are many ways to combine the results of a set of classifiers, depending on the type of the classifiers' output [7, 28]. If the output is only the best ranked class then majority voting can be applied. Some clas-

[1] To compute the contour direction, the windows to the left and to the right of the actual window are used.

sifiers have a ranked list of classes as output. In these cases often Borda count [12] or related methods are used. In the most general case, a classifier generates a score value for each class. In this case the sum, product, maximum, minimum, or the median of the scores of all classifiers can be calculated and the class with the highest value is regarded as the combined result [18]. Another approach is to use the score values output by the individual classifiers as input for a trainable classifier, e.g. a neural-network [32], which acts as the combiner.

Two combination schemes used in this paper are based on majority voting. In majority voting the top choice of each classifier is considered. The word class that is most often on the first rank is the output of the combined classifier. The two voting combination schemes differ in how ties are handled:

- Voting maximum score (*v_score*): Ties are broken by means of the maximum rule, which is only applied to the competing word classes. The maximum rule decides for the word class with the highest score among all word classes and all classifiers.

- Voting priority (*v_priority)*: In case of a tie always the output of a predefined classifier is used. Often the best performing classifier is chosen.

Two other combination schemes were used in the experiments:

- Weighted voting (*perf_v*): Here we consider again the top class of each classifier. In contrast with regular voting, a weight is assigned to each classifier. The weight is equal to the classifier's performance (i.e. recognition rate) on the training set. The output of the combined classifier is the word class that receives the largest sum of weights.

- GA weighted voting (*ga_v*): This combination scheme is similar to weighted voting, but the optimal weights are calculated by a genetic algorithm based on the results of the classifiers achieved on the training set. In [11] a detailed description of the scheme can be found.

It should be noted that for the case where the ensemble consists of three classifiers the schemes *perf_v* and *ga_v* usually are equivalent to the *v_priority* scheme. Therefore the *perf_v* and *ga_v* schemes will not be used to combine three classifiers.

4 Creation of an ensemble from a set of base classifiers

The method used in this paper for creating an ensemble from a set of base classifiers was introduced in [9] and is shortly described in this section. The underlying idea is very simple. Rather than starting with a single classifier, as it is done, for example, in Bagging [3], AdaBoost [8] and the random subspace method [13], we initially consider a set of classifiers (called prototypes in the following) and use an ensemble method to generate an ensemble out of each individual base classifier. Then we merge all classifiers of these ensembles to get a single ensemble. This procedure is more formally described in Table 1. For further details see [9]. Please note that the algorithm needs an underlying ensemble method, EM. The EMs used in the experiments are Bagging [3] and AdaBoost [8].

A well performing ensemble is characterized by two key properties. First the classifiers of the ensemble are diverse and secondly, the individual classifiers have a good performance. The goal of any ensemble method is to produce ensembles with both properties. An ensemble contains diverse classifiers if the misclassification of patterns has a low correlation across different classifiers (or in other words, the recognition rate of a classifier C_i on the patterns misclassified by another classifier C_j should be close to the average recognition rate of C_i). In the ideal case independent classifiers are created, but this is almost impossible in real world applications. The proposed ensemble method is expected to produce diverse classifiers as the classifiers are created from different base classifiers.

5 Experiments

For isolated character and digit recognition, a number of commonly used databases exist. However, for the task considered in this paper, there exists only one suitable database to the knowledge of the authors, holding a sufficiently large number of words produced by different writers, the IAM database [22]. Consequently this database was used in the experiments.

The training set contains 18,920 words and a large test set of 3,264 words was used. The vocabulary of the experiments contains 3,997 words, i.e. a classification problem with 3,997 different classes is considered. The set of writers of the training set and the set of writers of the test set are disjoint, so the experiments are writer independent. The total number of

Input: base classifiers $C_1, \ldots, C_n$; ensemble method EM; number of classifiers per ensemble, m.
Output: $m \cdot n$ classifiers.
CS is an empty set of classifiers;
for(i:=1;i<=n;i++)
    use EM with C_i as base classifier to produce m classifiers;
    put all produced classifiers in CS;
return CS;

Table 1. Ensemble method starting with a set of base classifiers

writers who contributed to this data set is 153, 116 for the training set and 37 for the test set.

First the three classifiers described in Section 2 were combined. The recognition rate of the base classifiers was 80.48 % for C_1, 71.57 % for C_2, and 78.65 % for C_3. The results of the combination are shown in Table 2. The combination with *v_priority* (C_1) produced the best result. This is not surprising as C_1 is clearly the best base classifier. The recognition rate of the multiple classifier system is better than classifier C_1 by 2.98 %.

The results of the novel ensemble method described in Section 4 are shown in Table 4. The entries in column *method* define the underlying ensemble method. The number of produced classifiers is given in column *size*. For each of the base classifiers $\frac{size}{3}$ classifiers were produced. For comparison purpose the results of the classical ensemble methods using the best base classifier, C_1, are given in the Table 3.

First it should be noted that both classical ensemble methods, whose results are reported in Table 3, improved the performance of base classifier C_1. In the best case the performance of the base classifier was increase by 1.54 %. Looking at Table 4, it is obvious that the ensemble method using all three base classifiers did much better than the classical ensemble methods. The recognition rate of the new ensemble method was also higher than the recognition rate obtained when combining the three classifiers (compare Table 2).

In Table 4 we observe that the *ga_v* combination doesn't reach the performance of the two other combination methods. A possible reason for this observation is that *ga_v* works on the training set to derive the weights of the individual classifiers. The strength of the overfitting for the base classifiers C_1, C_2 and C_3 is very different. C_2 overfits rather strongly, but C_3 doesn't overfit much. The overfitting behavior of the different classifiers may bias the combination, i.e. the weights of the classifiers produced from C_2 are likely to be too high. The problem could be solved by using an independent validation set to calculate the weights.

method	v_score	perf_v	ga_v
Bagging	0.05 %	0.05 %	13 %
AdaBoost	0.05 %	0.05 %	22 %

Table 5. Significance level of the superiority of the ensemble method using three base classifiers over the combination of the three classifiers with v_priority (C_1). (The null hypothesis is that both methods have identical performance.)

To compare the new ensemble method more thoroughly to the combination of the three classifiers, a statistical analysis was done. It was checked how significant the superior performance of the ensemble method is when compared to the combination of three classifiers with *v_priority* (C_1). The results are shown in Table 5. The low significance levels for the two combination schemes *v_score* and *perf_v* indicate that the superiority of the ensemble method is no coincidence.

Finally the diversity of the ensembles produced in the experiments was measured. Those diversity values are shown in Table 6. Also the average recognition rate of the classifiers was calculated. The diversity is defined as the average percentage of different results when comparing two classifiers of the ensemble. The diversity of the ensemble consisting of the three base classifiers is the highest. In the case of the ensemble methods using only one base classifier the diversity is quite low which lead to only moderate increases of the performance over the base classifier. The classifiers produced by the new ensemble method have a quite low performance, but the high diversity lead to well performing ensembles (compare Table 4).

6 Conclusions

Three handwritten word classifiers were introduced in the paper. It was shown that by combining the three classifiers the performance can be increased

v_score	v_priority (C_1)	v_priority (C_2)	v_priority (C_3)
83.36 %	83.46 %	81.83 %	82.75 %

Table 2. Results of the combination of the three classifiers with different architectures. The recognition rate of the base classifiers was 80.48 % for C_1, 71.57 % for C_2, and 78.65 % for C_3.

method	size	v_score	perf_v	ga_v
Bagging (only C_1)	21	81.1 %	80.91 %	81.13 %
AdaBoost (only C_1)	14	82.02 %	81.89 %	81.92 %

Table 3. Results of the classical ensemble methods using base classifier C_1. The recognition rate of base classifier C_1 is 80.48 %.

method	rec. rate	diversity
C_1, C_2, C_3	76.45 %	30.23 %
Bagging (only C_1)	78.7 %	14.39 %
AdaBoost (only C_1)	78.41 %	17.73 %
Bagging (C_1, C_2, C_3)	75.93 %	25.55 %
AdaBoost (C_1, C_2, C_3)	75.48 %	27.52 %

Table 6. Average recognition rate (*rec. rate*) and the diversity (*diversity*) of the classifiers in the produced ensembles.

over the best single classifier. An increase of 2.98% was obtained with the best combination scheme. By using classical ensemble methods, such as Bagging and AdaBoost, the performance could also be increased. The best performance was achieved with a new ensemble method proposed by the authors. The new method is distinguished from classical ensemble methods by the fact that is uses several base classifiers, rather than just a single one, to derive an ensemble. The performance of the new ensemble method was 1.5 % higher than the best combination of the base classifiers, 2.94 % higher than the classical ensemble methods, and 4.48 % higher than the best base classifier.

Future research will focus on the development of additional base classifiers and the evaluation of the proposed ensemble method for more than three classifiers. Also the relationship between performance and ensemble size may be addressed in future research.

Acknowledgment

This research was supported by the Swiss National Science Foundation (Nr. 20-52087.97). The authors thank Dr. Urs-Victor Marti and Dr. Alessandro Vinciarelli for providing the handwritten word recognizers and Matthias Zimmermann for the segmentation of a part of the IAM database. Additional funding was provided by the Swiss National Science Foundation NCCR program "Interactive Multimodal Information Management (IM)2" in the Individual Project "Scene Analysis".

References

[1] *Proc of the 7th Int. Conf. on Document Analysis and Recognition*, Edinburgh, Scotland, 2003.

[2] A. Brakensiek, J. Rottland, A. Kosmala, and G. Rigoll. Off-line handwriting recognition using various hybrid modeling techniques and character n-grams. In *7th International Workshop on Frontiers in Handwritten Recognition*, pages 343–352, 2000.

[3] L. Breiman. Bagging predictors. *Machine Learning*, (2):123–140, 1996.

[4] T. Dietterich and E. Kong. Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. Technical report, Departement of Computer Science, Oregon State University, 1995.

[5] T. G. Dietterich. Ensemble methods in machine learning. In *[19]*, pages 1–15.

[6] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artifical Intelligence Research*, 2:263–286, 1995.

[7] R. Duin and D. Tax. Experiments with classifier combination rules. In *[19]*, pages 16–29.

[8] Y. Freund and R. E. Schapire. A decision-theoretic generalisation of on-line learning and an application to boosting. *Journal of Computer and Systems Sciences*, 55(1):119–139, 1997.

[9] S. Günter and H. Bunke. Generating classifier ensembles from multiple prototypes and its application to handwriting recognition. In *[20]*, pages 179–188.

[10] S. Günter and H. Bunke. Optimizing the number of states, training iterations and Gaussians in an

method	size	v_score	perf_v	ga_v
Bagging (C_1, C_2, C_3)	21	84.83 %	84.96 %	83.92 %
AdaBoost (C_1, C_2, C_3)	15	84.96 %	84.93 %	83.85 %

Table 4. Results of the ensemble method using several base classifiers.

HMM-based handwritten word recognizer. In *[1]*, volume 1, pages 472–476.

[11] S. Günter and H. Bunke. Optimization of weights in a multiple classifier handwritten word recognition system using a genetic algorithm. *Electronic Letters of Computer Vision and Image Analysis, ELCVIA,*, 1(25-44), January 2004.

[12] T. Ho, J. Hull, and S. Srihari. Decision combination in multiple classifier systems. *IEEE Trans. PAMI*, 16:66–75, 1994.

[13] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[14] S. Impedovo, P. Wang, and H. Bunke, editors. *Automatic Bankcheck Processing*. World Scientific Publ. Co, Singapore, 1997.

[15] A. Kaltenmeier, T. Caesar, J. Gloger, and E. Mandler. Sophisticated topology of hidden Markov models for cursive script recognition. In *Proc. of the 2nd Int. Conf. on Document Analysis and Recognition, Tsukuba Science City, Japan*, pages 139–142, 1993.

[16] G. Kim, V. Govindaraju, and S. Srihari. Architecture for handwritten text recognition systems. In S.-W. Lee, editor, *Advances in Handwriting Recognition*, pages 163–172. World Scientific Publ. Co., 1999.

[17] M. Kirby. *Geometric Data Analysis: An Empirical Approach to Dimensionality Reduction and the Study of Patterns*. John Wiley and Sons, New York, 2001.

[18] J. Kittler, R. Duin, and M. Hatef. On combining classifiers. *IEEE Trans. PAMI*, 20:226–239, 1998.

[19] J. Kittler and F. Roli, editors. *First International Workshop on Multiple Classifier Systems*, Cagliari, Italy, 2000. Springer.

[20] J. Kittler and F. Roli, editors. *Third International Workshop on Multiple Classifier Systems*, Cagliari, Italy, 2002. Springer.

[21] D. Lee and S. Srihari. Handprinted digit recognition: A comparison of algorithms. In *Third International Workshop on Frontiers in Handwriting Recognition*, pages 153–162, 1993.

[22] U. Marti and H. Bunke. The IAM-database: An English sentence database for off-line handwriting recognition. *Int. Journal of Document Analysis and Recognition*, 5:39–46, 2002.

[23] U.-V. Marti and H. Bunke. Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system. *Int. Journal of Pattern Recognition and Art. Intelligence*, 15:65–90, 2001.

[24] H. Nishimura, M. Kobayashi, M. Maruyama, and Y. Nakano. Off-line character recognition using HMM by multiple directional feature extraction and voting with bagging algorithm. In *5th International Conference on Document Analysis and Recognition*, pages 49–52, Bangalore, India, 1999.

[25] D. Partridge and W. B. Yates. Engineering multiversion neural-net systems. *Neural Computation*, 8(4):869–893, 1996.

[26] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.

[27] J.-C. Simon. Off-line cursive word recognition. *Special Issue of Proc. of the IEEE*, 80(7):1150–1161, July 1992.

[28] C. Suen and L. Lam. Multiple classifier combination methodologies for different output level. In *[19]*, pages 52–66.

[29] C. Suen, C. Nadal, R. Legault, T. Mai, and L. Lam. Computer recognition of unconstrained handwritten numerals. *Special Issue of Proc. of the IEEE*, 80(7):1162–1180, 1992.

[30] A. Vinciarelli. *Offline Cursive Handwriting: From Word to Text Recognition*. PhD thesis, University of Bern, Switzerland, 2003.

[31] A. Vinciarelli, S. Bengio, and H. Bunke. Offline recognition of large vocabulary cursive handwritten text. In *[1]*, volume 2, pages 1101–1105.

[32] C. Wenzel, S. Baumann, and T. Jager. Advances in document classification by voting of competitive approaches. In J. Hull and S. Taylor, editors, *Document Analysis System 2*, pages 385–405. World Scientific, 1998.

[33] T. Windeatt and F. Roli, editors. *4th Int. Workshop on Multiple Classifier Systems*, Guildford, United Kingdom, 2003. Springer.

[34] L. Xu, A. Krzyzak, and C. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 22(3):418–435, 1992.

[35] S. J. Young, J. Jansen, J. J. Odell, D. Ollason, and P. C. Woodland. *The HTK Hidden Markov Model Toolkit Book*. Entropic Cambridge Research Laboratory, http://htk.eng.cam.ac.uk/, 1995.

[36] M. Zimmermann and H. Bunke. Hidden Markov model length optimization for handwriting recognition systems. In *Proc. of the 8th International Workshop on Frontiers in Handwriting Recognition*, pages 369–374, 2002.