On-Line Digit Recognition using Off-Line Features

A.Teredesai[1], E.Ratzlaff[2], J.Subrahmonia[2], V.Govindaraju[3]
1)Department of Computer Science,
Rochester Institute of Technology, Rochester, NY 14623, USA
2)IBM T.J.Watson Research Center,
Yorktown Heights, NY 10598, USA
3)Department of Computer Science and Engineering,
Center of Excellence for Document Analysis and Recognition,
State University of New York at Buffalo, Buffalo, NY 14260, USA
amt@cs.rit.edu, (ratzlaff,jays)@us.ibm.com, govind@cedar.buffalo.edu

Abstract

This paper describes a classification method for on-line handwritten digits based on off-line image representations. The goal is to use image-based features to improve classifier accuracy for on-line handwritten input. In this paper we describe an initial framework that can be used to achieve this goal. This framework for handwritten digit classification is based on genetic programming (GP). Several issues in preprocessing, transformation of data from on-line to off-line domains and feature extraction are described. Results are reported on the UNIPEN digit dataset.

1. Introduction

One of the fundamental aspects of handwritten character recognition methodologies is the manner in which data is collected. The data acquisition process can be on-line or off-line. On-line handwritten data is collected using a digitizer or an instrumented pen to capture the pen-tip position (x_t, y_t) as a function of time. Contrary to on-line, off-line handwritten data is collected using a scanner resulting in the generation of the signal as an image $I(x, y)$. Both areas of handwritten character recognition have been in existence for more than three decades. Various surveys outline the state of the art in each area [17] [1].

In this paper we present an implementation for on-line handwritten digit recognition using hierarchical feature extraction previously used in off-line handwritten digit recognition [15]. The focus is to describe how a combination of off-line features extracted from the on-line data and presented to GP [6] classifiers can provide good recognition results. We demonstrate this by using hierarchical feature extraction [9] and active digit recognition [15] to develop classifiers for the on-line UNIPEN digit dataset.

There were two main considerations for the use of off-line features in our implementation: a) Hierarchical feature space has proved to be very effective for digit classification and pairwise discrimination of confusing allographs [9] and b) Genetic Programming based classification has helped identify the important features for classifier development. Specifically, GP has several advantages over classical feature selection techniques when used for off-line handwritten digit classification using hierarchical features [15, 16]. Our aim here is to demonstrate that these conjectures are also true for on-line digit recognition. The approach we take here is to transform the data from the on-line signal domain to the off-line image domain and perform GP based classification.

As shown in Figure 1 the pen-down stroke is isolated in the UNIPEN data format as a first stage of preprocessing. This stroke information is converted into a Postscript image. We experimented with various parameters like resampling rate, thickness for painting the pen-down stroke, size of the image, etc. and used the most suitable representation. This Postscript image is then converted into a HIPS [8] image for simplified processing and hierarchical feature extraction. Feature vectors are then input to the GP system for classifier development.

This paper is organized in 5 sections. A discussion on key issues in on-line digit classification is presented in the next section. Section 3 outlines the GP based method for off-line handwritten digit classification. Section 4 provides a short introduction to classification using genetic programming. Section 5 discusses the experiments and results.

2. Issues in On-Line Classification

Isolated-characters and word recognizers are the most basic type of classifiers currently used in handwriting recognition. If the basic ink object under consideration is a single character or gesture the classifier is considered a character

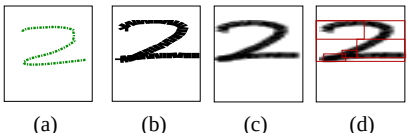

Figure 1: Extracting off-line hierarchical features from on-line data. As shown in the figure (a) on-line stroke signal is re-sampled for uniformity and isolated in a single file. (b) Postscript image generated from the isolated stroke. (c) HIPS image after conversion from Postscript. (d) Feature extraction stage dividing the input image into regions using a quad tree.

classifier. Complex word or glyph (multi-character shapes or fused characters) classifiers are often termed 'recognizers'. All these flavors of data complexity are present in the UNIPEN project dataset. The UNIPEN [5] project is an effort for benchmarking handwriting recognition systems. It consists of *Dataset(1a)- Isolated Digits* to *Dataset(8)- Free Text and Full Characters*. The scope of the current discussion is the development of a digit classifier for the 1a set.

The factors that influence classification of characters in a pen-based input system are geometric variations, stroke formation speed and allographic variations pertaining to positioning of the character and sequencing of strokes during inking. Plamondon et.al. [11] provide a more detailed discussion on each of these factors. Schomaker [13] notes the importance of usability of character classification in context to free-form text. The problems associated with achieving high accuracy at the isolated character level have led to the use of special symbols (e.g. Graffiti) in several commercial implementations. The development of classifiers is attempted using a variety of techniques ranging from structural and rule-based methods to statistical modeling. Rule-based methods do not require a large amount of training data and the number of features used to describe one class may be different from those required for another class. Disambiguating features like the difference in the direction of the stroke formation when writing the '-' of a '2' and the lack of a '-' in the lower portion of a '7' can be treated as rules to effectively assign class labels in pair-wise decisions. The performance of rule-based methods is limited by the capabilities of the designer to reliably devise the set of rules. On the other hand, statistical approaches generally require a large amount of data for training. Such classifiers require a fixed number of features in multidimensional feature space. The problem now is to define a separation boundary between classes in this feature space. This often leads to complex solution representations and intolerance to noise and consequently large generalization errors. As a result heavy emphasis is placed on the preprocessing stage prior to classification in order to effectively reduce these problems.

The motivation in this paper is to effectively blend the simplicity and speed of rule-based methods and the generalization power of statistical approaches in a unified framework using off-line image representation and GP. The combination of rule-based methods and learning is achieved using GP where solutions are evaluated using rules and new solutions are generated using statistical learning [6]. The conversion from on-line to off-line domain and the resulting advantages and disadvantages are discussed in the next section.

3. Issues in Off-Line Classification

In the domain of off-line handwritten character recognition there are two key strategies in current use. They can be broadly grouped as 'active' and 'passive' character recognition. Passive character recognizers use methods such as those described by Favata [4] and Suen [14]. These classical methods employ a *One Model Fits all* [10] paradigm since they assume all features to be equally important and base the decision on the entire feature vector. Active methods on the other hand try to perform feature selection and classification based on the nature (complexity) of the task at hand. Hence, active classifiers generated (using GP) use the multi-resolution feature space more optimally. Some systems, like the adaptive resonance network, learn as they perform [3] but lack feedback mechanisms to describe the learning strategy. Active character recognition as described by J.Park et.al. [9] [10] employs an active heuristic function that adaptively determines the length of the feature vector as well as the features themselves used to classify an input pattern. A feedback mechanism (rules) to understand why certain features are used by the classifier for a particular digit can be derived from the GP classifier tree. In terms of

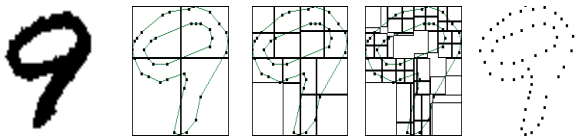

Figure 2: The preprocessing and feature extraction using a Quad tree [9]. The first figure shows the handwritten digit '9'. The second figure shows the normalized image segregated into 4 sub-images and the remaining figures show further segregation to derive a gradually detailed representation in feature space. The fifth figure shows the critical points after feature extraction.

GP classifier modeling strategies for handwritten character recognition several techniques have been implemented [15]. The theory and trade-offs in using GP have been discussed

by Koza et.al [7]. The solution representation in our approach is less complex compared to previously described methods. GP-trees generated as classifiers provide an insight into which features were used more often than others. In a hierarchical feature space these features can be mapped back to the regions of the image from which they were extracted. This in turn helps us analyze and interpret the classification results more effectively compared to currently used techniques [15]. At the heart of our approach is the feature extraction routine that divides the image into a quad tree. Features are then extracted based on the contour representation in the sub-image. Hierarchy is maintained by dividing each sub-image into deeper quad trees in order to increase the level of detail. For most datasets a depth of 4 is sufficient. The features from all these levels are then placed in a feature vector. This feature vector is then presented to the GP classifier. The same technique can be adopted using quin tree representation. This process of feature extraction and pre-processing is shown in Figure 2.

4. Genetic Programming (GP) Classifiers

GP has been used over the past couple of years in several domains, primarily because of its ability to model problems effectively. The way GP-trees are grown and evolved can be observed by a simple example [2] given in Figure 3. GP consists of representing solutions as programs that can be evaluated. Consider a GP-tree representation of the function $((x * x)/2.0)$ shown in Figure 3. The internal nodes are the operators '/' and '*' and the leaf nodes are x and 2.0. The problem at hand is easily represented using such expression trees. Inputs to the problem are represented as leaf nodes and are called terminals. Any operator used to operate on terminals is called a function. Functions and terminals together form a solution. Solutions are evaluated by instantiating the terminals and evaluating the functions. As solutions are represented with terminals and functions, the data types and length of these entities can vary from the simplest (e.g. integer) to more complex user defined types (e.g. Structures in C language). Thus, solutions are logically referred to as programs and the search is based on genetic methods [6]. The search for the best solution to solve a problem emulates the evolutionary process. An initial population of solutions is randomly generated and evaluated. Best individual solutions are selected for mating based on values obtained from fitness functions and new solutions are created using genetic operators. Fitness functions essentially evaluate a solution's ability to solve a problem. The new population is then evaluated and the search for a better solution continues until an optimal solution is found or other user-defined termination criteria are met. Keeping in view the above procedure to generate GP

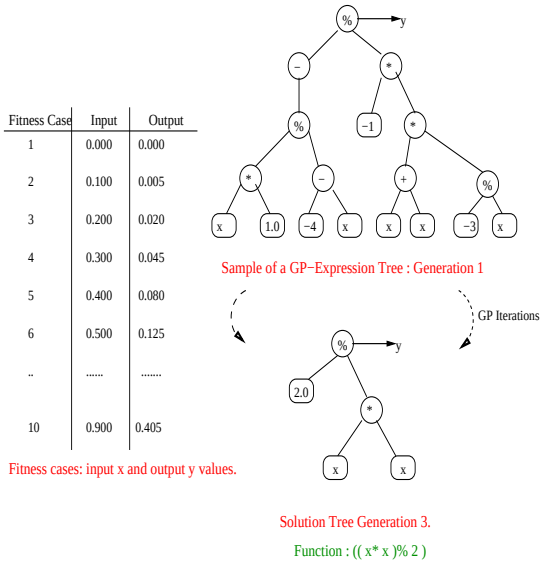

Figure 3: Example showing modeling of a GP solution [2]. The inputs and expected outputs are given in the table (bottom-left). A sample tree randomly initialized in the first generation is shown (top). Search for the best equation to represent the inputs and expected outputs leads to the solution tree representing function $((x * x)/2)$.

solutions, we initialize a population of GP trees using a set of functions which act as operators and a set of terminals which are subsets of the feature vector. Each tree therefore is a Lisp S-expression consisting of operators as internal nodes and features as terminal nodes. Feature vectors are computed from the training data and stored in a flat file. Each GP tree expression is then evaluated against each feature vector corresponding to a sample in the training data. The result of each evaluation is mapped to a real value that expresses a detection level for the sample. A detection level above +1 is positive and a level below -1 is negative. Any value in between is considered uncertain. Since there are 10 classes in the UNIPEN 1a digit set, we train 10 detectors. Each detector is trained to recognize one particular class of input. The detectors are then combined in parallel to form an 11-class (0-9 and reject) classifier and are evaluated against the validation set. Depending on the performance on the validation set, individual detectors undergo further training if required. The fitness of each i^{th} detector X_i is computed based on the equation : $F(X_i) = (N_{ir} * P_{ir}) + (N_{C_i} * P_{C_i}) + (N_{i\Phi} * P_{i\Phi}) + P_{ij}$ where, N_r = number of samples rejected due to output being between +1 and -1. P_r = penalty for rejection. N_{C_i} = number of training set samples not returning positive that belong to the class of the detector. P_{C_i} = penalty associated

with wrong classification of the sample. N_Φ= number of training set samples that returned a false positive and P_Φ= penalty associated with false positive classification. J= the criterion deciding minimum number of samples of class C_i that must be targeted by every solution in order to maintain potency in the population. P_j= penalty associated with not meeting J. The value of J must be decided before we begin the run and should usually fall in the range of 1-5% of the number of samples belonging to class C_i in the training set. Typical values of penalty in each case can range between 1 and 5 in our implementation. The training proceeds in GP generations until the required accuracy is achieved or the number of allowed generations is exhausted.

5. Experiments and Results

We use the UNIPEN [5] Train-R01/V07 subset 1a dataset consisting of 15,953 digits. Samples of some digits in this dataset are shown in Figure 4. These are the Postscript versions of the data converted from the UNIPEN format to Postscript using the utilities available as part of the UNIPEN Tools (http://unipen.nici.kun.nl/uptools3). As can be seen, the representation leads to wedges in the image. It has not been conclusively determined if such wedges affect the recognition accuracy in our experiments.

Figure 4: Different samples randomly selected from the UNIPEN 1a isolated digit set. These images are generated using the UNIPEN to Postscript conversion utility.

These Postscript images are then converted to the HIPS [8] format. There are several advantages to using the HIPS format including unrestricted image size and sequence length and the stored images are self-documenting. HIPS maintains a complete log of the file history within the header file. Every image file contains a record of who owns it, when it was created, the file format as well as a record of all HIPS operations performed on it which aids the feature extraction routine for hierarchical feature extraction as described by Park [9]. Features are extracted from these images by dividing each image into a quad tree with maximum depth of 4. At each level 8 features based on gradient and moment are computed. The total number of sub-images is $1 + 4 + 16 + 64 = 85$ thereby giving a feature vector of 680 features corresponding to each isolated digit in the dataset.

We decided to use 15,000 digits from the full set after some preliminary cleaning. The cleaning criterion eliminated erroneous data such as multi-digit entries or incomplete entries consisting of isolated strokes such as single dashes or dots. From the 15,000 digits 10,000 were used as training set, 2000 as validation set and 3000 as test set. We experimented with different training set sizes. Presenting smaller training sets led to more complex GP classifiers in terms of feature set size selected by GP. In retrospect, this led to higher confusion when the best of generation GP solution was evaluated on the validation set and we decided to present GP with more training data to capture the important variations more effectively in a more optimal feature subset. The truth information was gathered by reading the truth from the UNIPEN file.

Table 1: GP Parameters Used in Experiments.

Parameters	Values
Objective:	Evolve GP classifiers: 0-9
Terminal Set:	Hierarchical features
Function set:	+, -, *, DIV, SIN, COS, LOG, EXP
Population sizes used:	200, 400, 600
Crossover probability:	80%
Mutation probability:	20%
Selection:	Tournament selection (Size 7)
Termination criterion:	$Error \leq 5\%$
Maximum generations :	1000
Maximum depth:	17
Initialization Method:	Half-and-Half
Overall accuracy:	89%
Solution complexity :	High
Number of features selected by GP:	174 out of 279 total

The GP parameters used for these experiments are given in Table 1. We experimented with variations in the function set and population sizes and the optimal configuration of parameters is tabulated. These parameters are decided prior to training and remain consistent over the entire GP run.

Training consists of developing ten (0-9) single-digit detectors. The output of each detector, D, is either (a) an affirmative $(D > +1)$ indication that the input has been positively identified as a class member, (b) a negative $(D < -1)$ indication that the input is not a class member, or (c) an uncertain $(-1 <= D <= +1)$ indication that the detector was unable to make either a positive or negative determination.

Once all the single-digit detectors are trained, they are combined into a higher-level 11-class (0-9, reject) classifier. During test decode phase each input is presented to every detector.

- If only one detector claims the input to be from *its class* **and** if all the other detectors classify the input as *negative or uncertain* **and** the *truth value matches* the claim, accept the input as **correctly** classified.
- The input is classified **wrong** if *two or more* detectors claim the input *as their class* **or** *truth value differs* from any claims.
- The input is **rejected** if no detector provides an affirmative value.

Table 2: Results obtained on UNIPEN 1a digit set.

Class	No.of Samples	Correct	Incorrect	Reject	Correct(%)
0	250	221	9	20	88.4
1	250	215	5	30	86.0
2	250	227	11	12	90.8
3	250	235	7	8	94.0
4	250	213	11	26	85.2
5	250	205	33	12	82.0
6	250	221	18	11	88.4
7	250	219	22	9	87.6
8	250	224	9	17	89.6
9	250	215	6	29	86.0

The best individual digit classification accuracy for a single class was 94% for digit 3. Rejection rate for digit 3 was 3.2%. The lowest accuracy was 82% for digit 5. The maximum inter-class confusion was for digits 4 and 9. The overall accuracy on the test set was 87.8% while the overall error was 5.2%. Class-wise accuracy, error and rejection results are given in Table 2.

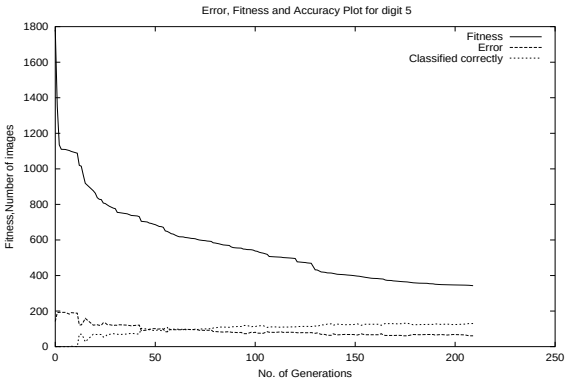

Figure 5: Plot showing fitness, error and accuracy over number of generations of a GP run during training of digit 5 detector. The curves shown in this plot are representative of most GP detectors where the fitness decreases sharply during the initial runs and stabilizes over the number of generations with small decrements. Note that in standard GP terminology lower fitness value represents better classification.

Compared to other state of the art on-line handwritten digit recognition technique the current recognition rates of this system are low. Ratzlaff [12], has reported a better performance using a simple scanning n-tuple classifier for the same dataset. He has also provided a comparative analysis of other competitive systems. Previously, GP based recognizers have been applied to the off-line handwritten digit recognition task [16] and the performance on standard NIST and USPS datasets is comparable to the state of the art in the off-line domain [15]. The aim here was to propose an alternative framework for on-line recognition based on off-line features and to present the issues faced in doing so. Efforts are on currently to improve the classification accuracy of the proposed system. The graph in Figure 5 shows the training curve of the detector for digit '5'. The error in training and fitness values are plotted. The initial search stabilizes after 250 generations, suggesting convergence.

6. Discussion

We described a new approach for isolated digit classification of UNIPEN data.The different approaches under consideration for future work are:

- Obtain the Fourier transform of the on-line signal and use the co-efficients as input features for GP based training.
- Use frame based features currently used for HMM based recognition.
- Effective combinations of the above.

This paper describes the approach to transform the data from on-line signal domain to off-line image domain and perform the GP based training. The classifiers work in the off-line image domain. Hierarchical feature extraction directly from original stroke vectors might prove a more robust approach. This will definitely reduce the dependence on converting data to Postscript and HIPS image representation thereby decreasing the noise.

The main objective here was to explore the possibilities of using off-line features to aid on-line recognition. GP was the technique of choice since we were familiar with GP performance in the off-line domain.

References

[1] N. Arica and F. Yarman-Vural. An overview of character recognition focused on off-line handwriting. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 31(2):216 – 233, May 2001.

[2] W. Benzhaf, P. Nordin, R. Keller, and F. Francone. *Introduction to Genetic Programming*. Morgan Kaufmann Publishers, 1998.

[3] G. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. In *Computer Vision, Graphics, and Image Processing*, pages 35–115, 1987.

[4] J. Favata and G. Srikantan. A multiple feature/resolution approach to handprinted digit and character recognition. *International Journal of Imageing Systems and Technology*, 7:304–311, 1996.

[5] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. Unipen project of on-line data exchange and recognizer benchmarks. In *Proceedings of the 12th International Conference on Pattern Recognition*, pages 29–33, Jerusalem, Israel, October 1994. IAPR-IEEE.

[6] J. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[7] J. Koza, F. Bennett, D. Andre, and M. Keane. *Genetic Programming III*. Morgan Kaufmann Publishers, 1999.

[8] M. Landy, Y. Cohen, and G. Sperling. Hips: Image processing under unix. *Behavior Research Methods, Instrumentation, and Computers*, 16:199–216, 1984.

[9] J. Park and V. Govindaraju. Active character recognition using a^**-like* algorithm. In *Proceedings of Computer Vision and Pattern Recognition*, 2000.

[10] J. Park, V. Govindaraju, and S. Srihari. *OCR* in a hierarchical feature space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(4):400–407, April 2000.

[11] R. Plamondon, D. Lopresti, L. Schomaker, and R. Srihari. Online handwriting recognition. *Wiley Encyclopedia of Electrical and Electronics Engineering*, 1999.

[12] E. Ratzlaff. A scanning n-tuple classifier for online recognition of handwritten digits. In L. S. K. Tombre and C. Fuh, editors, *Proceedings of The Sixth International Conference on Document Analysis and Recognition*, pages 18–22, September 2001.

[13] L. Schomaker. From handwriting analysis to pen-computer applications. *Electronics and Communication Engineering Journal*, pages 93–102, June 1998.

[14] C. Suen. Character recognition by computer and applications. In T.Y.Young and K. Fu, editors, *Handbook of Patter Recognition and Image Processing*, pages 569–589. Academic Press, 1986.

[15] A. Teredesai and V. Govindaraju. Active digit classifiers: A separability optimization approach to emulate cognition. In L. S. K. Tombre and C. Fuh, editors, *Proceedings of The Sixth International Conference on Document Analysis and Recognition*, pages 401–406, September 2001.

[16] A. Teredesai, J. Park, and V. Govindaraju. Active handwritten character recognition using genetic programming. In *EuroGP2001 Proceedings*, 2001.

[17] O. Trier, A. Jain, and T. Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, 29(4):641–662, 1996.