© IEE 1993 14 September 1993
Electronics Letters Online No: 19931348

M. Steyaert and J. Crols (*K.U.Leuven, ESAT-Micas, Kardinaal Mercierlaan 94, 3001 Heverlee, Belgium*)

References

1 ADACHI, T., ISHINAWA, A., BARLOW, A., and TAKASUKA, K.: 'A 1.4 switched capacitor filter'. Proc. CICC, May 1990, (Boston), pp. 8.2.1–8.2.4

2 CASTELLO, R., and TOMASINI, L.: '1.5-V high-performance SC filters in BiCMOS technology', *IEEE J. Solid-State Circuits*, 1991, **SC-26**, (7), pp. 930–936

3 CALIAS, F., SALCHI, F.H., and GIRARD, D.: 'A set of four IC's in CMOS technology for a programmable hearing aid', *IEEE J. Solid-State Circuits*, 1989, **SC-20**, (2), pp. 301–312

4 BECKER, R., and MULDER, J.: 'A low-power DTMF and signalling frequency detector'. Proc. ESSCIRC, September 1990, (Grenoble), pp. 5–8

5 HUIJSING, J.H., and LINEBARGER, D.: 'Low-voltage operational amplifier with rail-to-rail input and output ranges', *IEEE J. Solid-State Circuits*, 1985, **SC-20**, (6), pp. 1144–1150

Efficient use of training data in the *n*-tuple recognition method

R. Tarling and R. Rohwer

Indexing terms: Pattern recognition

A simple technique is presented for improving the robustness of the *n*-tuple recognition method against inauspicious choices of architectural parameters, guarding against the saturation problem, and improving the use of small data sets. Experiments are reported which confirm that the method significantly improves performance and reduces saturation in character recognition problems.

Introduction: The *n*-tuple recognition method of Bledsoe and Browning [1] often achieves accuracies competitive with the best methods available, while offering an overwhelming advantage in learning speed [2]. A simple technique is suggested for improving the robustness of the method against inauspicious choices of architectural parameters, guarding against the 'saturation' problem, and improving the use of small data sets. The technique has been tested in both a near real-time single-user character recognition system implemented on a PC with input via a digitiser [4], and a freely available multi-writer optical character recognition (OCR) database.

n-tuple method: The simplest variation of the *n*-tuple method was used. The patterns to be classified are bit strings of a given length. Several (let us say *N*) sets of *n* bit locations are selected randomly. These are the *n*-tuples. The restriction of a pattern to an *n*-tuple can be regarded as an *n*-bit number which constitutes a 'feature' of the pattern. A pattern is classified as belonging to the class for which it has the most features in common with at least one training pattern of that class.

Precisely, the class assigned to unclassified pattern *u* is

$$\underset{c}{\operatorname{argmax}} \left(\sum_{i=1}^{N} \Theta \left(\sum_{v \in C_c} \delta_{\alpha_i(u),\alpha_i(v)} \right) \right) \tag{1}$$

where C_c is the set of training patterns in class c, $\Theta(x) = 0$ for $x \leq 0$, $\Theta(x) = 1$ for $x > 0$, $\delta_{i,j}$ is the Kronecker delta ($\delta_{i,j} = 1$ if $i = j$ and 0 otherwise) and $\alpha_i(u)$ is the ith feature of pattern u:

$$\alpha_i(u) = \sum_{j=0}^{n-1} u_{\eta_i(j)} 2^j \tag{2}$$

Here u_i is the ith bit of u and $\eta_i(j)$ is the ith bit location of the ith n-tuple.

With C classes to distinguish, the system can be implemented as a network of NC nodes, each of which is a RAM. The memory content $m_{ci\alpha}$ at address α of the ith node allocated to class c is

$$m_{ci\alpha} = \Theta \left(\sum_{v \in C_c} \delta_{\alpha,\alpha_i(v)} \right) \tag{3}$$

Thus $m_{ci\alpha}$ is set if any pattern of C_c has feature α and unset otherwise. Recognition is accomplished by tallying the set bits in the nodes of each class at the addresses given by the features of the unclassified pattern.

The modification: Although it has much in common with other neural and statistical learning methods, this method departs from the common philosophy of fitting data to a model by minimising an error measure with respect to model parameters. Therefore it suffers relatively little from overfitting on small datasets. But whereas a minimisation-based method always benefits from an increased amount of training data, the *n*-tuple method can suffer from 'saturation'. The problem is that because bits in memory are set but never reset, the contents can tend toward solid ones, so the capacity for discrimination is lost. This problem will clearly be present if the amount of memory in each node (2^n bits) is too small for the amount and noisyness of the data. Performance can also be adversely affected by setting *n* too large [3], which may also require an impractical amount of memory.

To optimise the amount of training data presented to the system a simple 'test before train' heuristic was developed to restrict the sum in eqn. 3. A training pattern was used to adjust RAM contents only if it was not correctly classified by the system in its current state of training. This process was repeated by rescanning the data until no further adjustments were required. (One example from each class is presented, then another, etc.) No more than three passes over the data were ever required in practice.

This method cannot be expected to help if *n* is chosen too large, but can prevent saturation if *n* is too small. Furthermore, it allows the system scope to adjust more closely to the structure of the data, in which relatively few or many features may distinguish different pairs of classes. More examples of classes having a large variance over the feature set would be selected than would be for classes concentrated on a few features.

The data: Two sources of data were used to test the method:

(i) The PC/digitiser system was used to collect 125 sets of 10 samples of isolated handwritten digits {0,...,9} from one writer. These were represented as 8 × 8 pixel arrays. 90 of the sets were randomly selected for training and the remaining 35 were used for testing.

(ii) The f13 subset of NIST Special Database 3 was also used. The full database [Note 1] includes 223125 digits handwritten by 2100 US census workers from throughout the USA, represented as 32 × 32 pixel arrays. The f13 subset [Note 2] contains a total of at least 140 samples of each digit distributed unevenly over 49 of these writers. 100 samples of each digit were randomly selected from the first 140 for training, and the remaining 40 used for testing.

On the single-writer data, a system of 4096 4-tuples improved insignificantly from 99.2% ±0.3% to 99.4% ±0.3% recognition accuracy when the test-before-train principle was applied, but saturation (the percentage of ones in memory) was reduced from 48.1% ±0.3% to 26.6% ±0.7%. These figures are means and standard deviations over 10 runs using different random mappings η in eqn. 2. The system required an average of only 10 examples from the 90 presented in order to capture the features of each digit.

On the NIST f13 data, the same 4-tuple system improved quite significantly from 84.7% ±0.9% recognition accuracy to 89.4% ±1.6% in one pass of the test-before-train method, and then to 91.9% ±0.9% on a second pass, after which no further improvement was possible. Saturation was lowered from 51.39%±0.05% to 38.39% ±0.28%, and an average of 44 of the 100 available training patterns were used.

The full NIST dataset was studied by 26 organisations with a total of 118 OCR systems represented at the First Census Optical

Note 1: for sale on CD-ROM from Standard Reference Data, National Institute of Standards and Technology, 221/A323, Gaithersburg, MD, 20899, USA

Note 2: freely available by anonymous ftp from sequoyah.ncsl.nist.gov.

Character Recognition Systems Conference [5]. Typical recognition rates were around 95% on the digits, with only one system (based on a set of multilayer perceptrons) approaching the human performance of 98.5%. For a system using less than 0.05% of the training data, the n-tuple method seems quite respectable. Extension of these preliminary tests to the full dataset would be an interesting project.

Similar tests were run using various tuple sizes from 2 to 10, with qualitatively similar results. Recognition accuracy is poor for 2-tuples, best for 4-tuples, and degrades slightly as the size increases beyond 4.

Conclusions: These results demonstrate in character recognition applications that the test-before-train heuristic provides a convenient way to control saturation in an n-tuple recogniser, thereby making efficient use of the available training data.

20 September 1993
Electronics Letters Online No: 19931390

R. Tarling (*92 Elmcroft Ave., Wanstead, London E11 2DB, United Kingdom*)

R. Rohwer (*Dept. of Computer Science and Applied Mathematics, Aston University, Aston Triangle, Birmingham B4 7ET, United Kingdom*)

References

1 BLEDSOE, W.W., and BROWNING, I.: 'Pattern recognition and reading by machine'. Proc. Eastern Joint Computer Conf., 1959, (Boston), pp. 232–255

2 ROHWER, R., and CRESSY, D.: 'Phoneme classification by Boolean networks'. Proc. European Conf. on Speech Communication and Technology, 1989, (Paris), pp. 557–560

3 ROHWER, R., and LAMB, A.: 'An exploration of the effect of super large n-tuples on single layer ramnets', in ALLINSON, N., Ed. 'Proceedings of the weightless neural network workshop '93, Computing with logical neurons' (University of York, 1993), pp. 33–37

4 TARLING, R.: 'Computer recognition of hand-printed characters using weightless neural networks'. Final year project report, Dept. of Computer Science and Applied Mathematics, Aston University, Birmingham, UK, 1993

5 WILKINSON, R., GEIST, J., JANET, S., GROTHER, P., BURGES, C., CREECY, R. HAMMOND, R., HULL, J., LARSEN, N., VOGL, T., and WILSON, C.: 'The first census optical character recognition systems conference'. Technical Report NISTIR 4912, National Institute of Standards and Technology, Gaithersburg, MD, USA, 1992

Digital signature with (*t, n*) shared verification based on discrete logarithms

L. Harn

Indexing terms: Information theory, Public-key cryptography

The Letter presents a digital signature scheme based on the discrete logarithm problem which enables any t of the n verifiers to verify the validity of the signature.

Introduction: The digital signature with (t, n) shared verification is the same as regular digital signatures which consists of a string of binary numbers generated by a single user with the knowledge of a secret key, except that the signature verification has the following properties:

(i) any t of the n verifiers can verify the validity of the signature

(ii) any t–1 or fewer verifiers cannot verify the validity of the signature.

This definition is very similar to the definition of a (t, n) secret sharing scheme. However, the major differences are:

(i) in the secret sharing scheme, because the secret shadows are exchanged among users and the master key is derived after each secret reconstruction process, the master key can only be used once if no other encryption scheme has been used; but, in a shared verification signature scheme, because the secret shadows and the master key are used to verify the signature, it may never be revealed in the cleartext form and thus the master key and secret shadows can be used repeatedly

(ii) in the shared verification signature scheme, the master key corresponds to the public key used in the digital signature schemes and by knowing only the public key, it is still infeasible to obtain the secret key used to sign the message.

Soete *et al.* [1] proposed a (t, n) shared verification signature scheme in 1989 based on generalised quadrangles. It requires the use of secure boxes for the verifiers to verify the signature. The possible applications of this signature scheme can be found in [1,2]. In this Letter, we propose a signature scheme with (t, n) shared verification based on the computational difficulty of discrete logarithms.

Proposed signature scheme with (t, n) shared verification: p = a prime modulus where $2^{511} < \mathrm{p} < 2^{512}$; $w = (p - 1)/2$, a large prime, where $2^{510} < \mathrm{w} < 2^{511}$ q = a prime divisor of $w - 1$, where $2^{159} < q < 2^{160}$; s = a secret integer for the user with $0 < s < q$; $y_v = \beta_v^s$ mod p, for $v = 1, 2, \ldots$, where y_v is the public key for the signer used for message m_v, and β_v is a generator with order w in $GF(p)$; $\{a_i$, for $i = 1, \ldots, t - 1\}$, and $f(x) = s + a_1x + \ldots + a_{t-1}x^{t-1}$ mod q, each a_i is a random integer with $0 < a_i < q$; $\{g_v$, for $v = 1, 2, \ldots\}$, where $g_v = h_v^{(p-1)/w}$ mod $p > 1$; each h_v is a random integer with $0 < h_v < p$; each g_v is a generator with order w in $GF(p)$, thus we have g_v^t mod $p = g_v^{t \bmod w}$ mod p for any non-negative integer t; $\alpha = e^{(w-1)/q}$ mod $w > 1$, e is a random integer with $0 < e < w$; α is a generator with order q in $GF(w)$, thus we have α^t mod $w = \alpha^{t \bmod q}$ mod w, for any non-negative integer t; m_v, for $v = 1, 2, \ldots$, are messages to be signed and transmitted; k_v, for $v = 1, 2, \ldots$, are random integers with $0 < k_v < w$; H = a one-way hash function.

Integers $\{a_i, i = 1, \ldots, t - 1\}$ are secret values, and p, w, q, and g_v, for $v = 1,2, \ldots$, are public values. The signer's private and public keys are s and y, for $v = 1, 2, \ldots$, respectively. s, α, and k_v must be kept secret. k_v must be changed for each signature.

Shadow generation: Our scheme uses the cryptographic techniques of the perfect secret sharing scheme of Shamir [3] based on the Lagrange interpolating polynomial and the digital signature algorithm [4] proposed by NIST.

Let A be the signer and s be the secret key used by A to sign messages. A is responsible for generating secret shadows for all verifiers. A selects the $(t - 1)$ th degree polynomial $f(x) = s + a_1 x + \ldots + a_{t-1}x^{t-1}$ mod q. The shadows for each verifier are computed as $S_i = \alpha^{f(x_i)}$ mod w, where x_i is the public information associated with the verifier u_i. We would like to point out here that with any t pairs of (x_i, S_i), α^s can be determined as $\alpha^s = \alpha^{f(0)}$ mod w

$$= \alpha^{\left(\sum_{i=1}^{t} f(x_i) \prod_{j=1,j\neq i}^{t} \frac{-x_j}{x_i - x_j} \bmod q\right)} \bmod w$$

$$= \prod_{i=1}^{t} S_i^{\left(\prod_{j=1,j\neq i}^{t} \frac{-x_j}{x_i - x_j} \bmod q\right)} \bmod w \qquad (1)$$

Signature generation: The signature scheme is based on the ElGamal signature scheme [5] with some modifications. Assume A wants to sign a message m_v, where $0 \leq m_v \leq p - 1$. With the knowledge of the secret key s, A can find β_v to satisfy the relation

$$g_v \alpha^s = \beta_v^s \bmod p \qquad (2)$$

We have defined $y_v = \beta_v^s$ mod p. A then randomly selects an integer k_v, where $0 \leq k_v \leq w - 1$, and computes

$$r_v = \beta_v^{k_v} \bmod p$$

A now solves the congruence

$$m_v' = k_v z_v + s r_v \bmod w$$

or

$$z_v = (m_v' - s r_v) k_v^{-1} \bmod w$$

for integer z_v, where $0 \leq z_v \leq w - 1$ and $m_v' = H(m_v)$. $\{z_v, r_v, g_v, \beta_v\}$ is the signature for message m_v.