International Journal of Computational Intelligence and Applications
Vol. 2, No. 2 (2002) 1–22

PASSPHRASE AUTHENTICATION BASED ON TYPING STYLE THROUGH AN ART 2 NEURAL NETWORK

JASON BECHTEL* and GURSEL SERPEN†
Electrical Engineering and Computer Science,
University of Toledo, Toledo, OH 43606, USA
**jason@uniqsys.com*
†*gserpen@eng.utoledo.edu*

MARCUS BROWN
Computer Science Department, University of Alabama,
Tuscaloosa, AL 35487, USA
mbrown@cs.ua.edu

Received 1 May 2002
Revised 17 June 2002

This study proposes the use of an artificial neural network algorithm to perform passphrase authentication based on the typing style of a user. The only hardware required is a keyboard. Prior studies have demonstrated the feasibility of this approach and its limitations, one of which was the need for collection of impostor samples for training the artificial neural network based classifier algorithm. This requirement is rather impractical for most application domains. The proposed study eliminates the need to collect impostor samples by employing an unsupervised and self-organizing artificial neural network algorithm, the Adaptive Resonance Theory 2 neural network, and therefore pushes the passphrase authentication technology one step closer to the realm of practical implementation. The preliminary study performed demonstrates that it is possible to train an Adaptive Resonance Theory 2 neural network using only authentic sample data and still provide a relatively low impostor pass rate. Given the minimal cost and easy in-field trainability of the proposed passphrase authentication system, the developed system can greatly enhance the security of computing environments with wide acceptance.

Keywords: Computer security; ART2 neural network; passphrase authentication; typing style; impostor pass rate.

1. Introduction

Computer security, at a comforting level of sophistication, is essential for the robust operation of e-commerce and e-society. Recent breaches in computer security and costs incurred as a result of these breaches are disproportionately significant. There is a justifiable perception on behalf of both the public at large and the government that computers need to be made more secure through urgent focused investments in

the field. Unauthorized access to computing systems is one of the primary sources of security breaches. Recent experiences prove current methods of simply verifying a passphrase against a stored passphrase for each user to be inadequate.

Computer systems enforce security by finding a way to draw a line between users who should have access to the system and intruders who should be denied access. The most pervasive form of computer security, the password, relies on secret knowledge to draw its line between users and intruders. This paradigm is inherently weak because password knowledge can be discovered or forced from a user or attacked with brute force by rapidly attempting many possibilities. Another widely employed authentication paradigm is that of the special access device. The users of the system carry the identification device on their person. This system relies on the ownership of the device and its mechanical or physical obscurity. It can be subverted by stealing or cloning a user's device.

Because of the inherent weaknesses in these security techniques, a vast amount of research and development has been invested in biometric authentication technologies. These paradigms exploit properties of the users themselves as authentication criteria. There are two kinds of biometric identification paradigms: behavioral and physical. Physical biometrics target quantifiable, unique physical features of the user. Examples of unique human features are fingerprints, facial appearance, blood-vessel arrangement in the face or the retina, and hand geometry. Behavioral biometrics target unique and highly consistent patterns of behavior. Examples of behavioral characteristics include voice, signature, and typing patterns.

Physical biometric authentication systems tend to require special-purpose hardware and software. The cost of the engineering and production of these systems drives the final prices upward. Behavior-oriented systems, on the other hand, match up nicely with existing computer interfaces and so require less special-purpose hardware and software. Voice pairs with microphones, signature with scanners and optical character recognition technology, and typing style with the ubiquitous keyboard interface. Behavior-based technologies are also subject to greater variability, however, and thus research is required to create stable, robust systems.

The objective of this study is to develop a practical, easy-to-deploy software-based passphrase authentication system that learns typing styles to distinguish users from impostors. The proposed system is designed such that no special hardware is required for the implementation of the authentication protocol. Furthermore, the software-based passphrase authentication system will be trainable to learn the idiosyncrasies of each user, will be adaptive to accommodate the variations in the typing styles of the users, and will not require impostor samples to be collected.

Work in the area of passphrase authentication through typing characteristics has existed for more than two decades. While the concept is not new, there is still uncertainty as to which methods are better and how to optimize the system. As the following paragraphs suggest, many studies have been conducted, but few with

a sufficient number of test cases to establish the reliability of the methods to a statistically significant degree.

The uniqueness of a user's typing pattern was reported by Joyce and Gupta as in the following excerpt: "The same neurophysiological factors that make a written signature unique are also exhibited in a user's typing pattern".[15] Based on the findings of neurophysiological studies, work has steadily progressed in using typing behavior as an authentication tool. Leggett, Williams and Usnick offer an overview of this work.[20] Each key pair is executed in one of three ways: one finger (1F), two fingers on one hand (2F), or two fingers on opposite hands (2H). Most of the variation appears to stem from the physiological constraints of 2F maneuvers. This variation persists regardless of typing speed among proficient typists. A study by Gentner provides evidence that the execution of typing actions is "programmed" in time periods and not in terms of actual finger movements and trajectories.[12] There is thus a solid foundation for the exploration of user identification based on typing characteristics.

Pioneering studies were performed by the United States Air Force[8] and the RAND Corporation,[9] which laid the foundation for further research by exploring the physiological underpinnings of typing behavior and by collecting large amounts of data.[20] Umphress and Williams[31] and Leggett and Williams[19] follow up on this work with experiments involving digraph latencies analyzed using statistical methods. The use of digraph latencies and statistical models was the popular approach for some time. The statistical methods were gradually modified and refined by Joyce and Gupta,[15] Leggett, Williams, and Usnick,[20] Napier *et al.*,[24] and Mahar *et al.*[23] Two patents were issued during this time for devices which use statistical approaches to user identification through typing characteristics.[10,32] The most recent work using statistical methods, sometimes referred to as "pattern recognition" techniques, uses a so-called "covariance–covariance" approach.[16]

Meanwhile, some were applying the relatively new field of artificial neural networks to the problem. Instead of statistical methods, which do not account very well for the many parameters that vary from individual to individual, these researchers hoped to hand off the difficult task of classification to a neural network. Brown and Rogers were among the first to attempt this and received a patent for their method.[2,3] They used a multi-layer perceptron (MLP) with back-propagation training (BP) and achieved a combined error rate of 11.5 percent. Unfortunately, the MLP/BP neural network carries the requirement of impostor (negative) samples during training, making it difficult to deploy in a practical setting. Lin extends work with the MLP/BP by examining variations on the structure and parameters of the neural network, eventually achieving a combined 1.1% error rate, with a relatively large test population.[21]

Obaidat and Sadoun performed a side-by-side comparison of five statistical classification methods and eight neural network paradigms. The results clearly favored neural networks over statistical methods in general. The best result of the statistical methods, achieved using the potential function algorithm, was a combined error

rate of 2.6%, while three of the neural networks were reported to have reached 100% accuracy. The results also showed the superiority of some neural network algorithms over others. Specifically, the fuzzy Adaptive Resonance Theory (ARTMAP) neural network was one of the three best algorithms with zero percent error. By altering the activation function for the hidden layer and the learning rule, Obaidat and Sadoun also achieved a 0.05% error rate with a MLP/BP neural network using a sigmoid transfer function. Yet, the MLP/BP with a sine-delta transfer function ranked as one of the worst classifiers, along with the Counter-Propagation neural network (CPNN), with approximately 26 and 56% combined error rates, respectively.[26]

The reported zero percent error rate of the fuzzy ARTMAP neural network for passphrase authentication in prior studies might not be representative of the true performance characteristics of the algorithm. Validation of those studies by other researchers, especially considering the small sample size of fifteen subjects, has not been reported. Other particulars of the Obaidat experiments may have contributed to the relative success of several neural network classifiers. One possible factor is the fine keystroke event capture resolution (0.0001 second). Another possible influence could be the large number of samples used. Each subject provided 225 samples per day for eight weeks.[25,26]

One of the algorithm-independent factors that can greatly affect classification results is the time resolution at which the keystroke events are monitored and recorded. If the resolution is too low, then the valuable information about the individual typist will not be recorded. Long *et al.* reported interkey latencies varying from 55 milliseconds (calculated from a typing speed of 216 words per minute) to hundreds of milliseconds ("hunt-and-peck" typists).[22] Many experimenters have arrived at an upper cut-off of 500 milliseconds, above which it is assumed that the typist has become distracted. Therefore, most research in this field has utilized millisecond resolution or better when recording keyboard events. Shepherd points out that, with a resolution of ten-thousandths of a second, keyboard events are recorded with an accuracy of approximately one percent.[29] Some researchers have expressed concern for the ability to reliably capture accurate timing data on time-sharing systems. Ostry, however, was using millisecond resolution with an IBM typewriter and a PDP-11 computer.[27] Most modern computing platforms should be able to provide at least millisecond resolution, and this is the recommended minimum for conducting such an experiment. Note, however, that some early work using statistical techniques was relatively successful with only centisecond resolution.[20,31]

When a user types proficiently, one key usually remains depressed when the next key is struck. This overlapping of keystrokes is known as a digraph and the two keys involved are the digraph pair. Almost all of the researchers in this area have focused their work on one or more aspects of the digraph event. Earlier work started by considering the digraph "latency", which is the time from the press of the first key to the press of the second key. Many of these earlier efforts focused on particular digraph pairs, which produced a desirable amount of variability, and only

considered those intervals. The main reason for this was to offset the weaknesses of the statistical approach to classification by eliminating as much noise as possible. Some then started to consider the key hold times as well, with improved results.[25] Eventually, both types of intervals were being used almost ubiquitously.[16,21,24,26,28] The results generally improved as more intervals were considered.[25,26] This may merely reflect the fact that enriching the information content of data can only aid in the classification process.

While the 1977 United States Air Force study found that names were too short for reliable identification, this is likely merely a reflection of the simplicity of the statistical methods employed to classify samples. The sample sizes in studies thereafter began high, with 1400 characters, then dropping down to 537 characters.[19] However, Garcia had already patented a device that worked using only the user's name.[10] Napier *et al.* demonstrated error rates of only 3.6% and 0.9% for samples of only 50 and 300 digraphs, respectively.[24] Finally, Obaidat[25] and Lin[21] achieved 0.0% and 1.1% error, respectively, using passphrases averaging only seven characters in length. Results generally show, however, that higher variability (in more difficult passphrases) leads to more false impostors.[4] Joyce and Gupta also assert that "well-known, regularly typed strings" provide the best consistency, and they used the first and last names of the participants as part of their protocol.[15] Still further evidence comes from the work of Brown and Rogers, who claim that familiar phrases, such as a user's name, will be more successful than passwords.[2]

All of the studies in this area of research attempt to achieve user identity authentication. Each must therefore account for the eventual scenario of an unknown user attempting to gain access to the computer system. The data from these attempts (test samples) must be either captured from actual users or somehow artificially generated. In the case of neural network classifiers, an additional requirement can be introduced. Some neural network algorithms require negative examples as part of the training data, or template. Researchers have mostly met this requirement by collecting additional typing samples from the experiment participants, both for their own and for other participants' passphrases. The precedent was set in the 1977 United States Air Force study.[8] One exception to this was a method for providing the impostor samples for the training of the multi-layer perceptron neural network with back-propagation training. Instead of participants typing the passwords of the other participants, authentic samples were multiplied by random values between 0.2 and 10 to develop an approximate model of impostor attempts.[21] Truly random variation is unrealistic, however, and therefore could lead to an artificially low impostor pass rate.

Another point of divergence from the general trend is to do what is called "bootstrapping" or "train and test" instead of collecting two sets of authentic data from each participant.[5,24] This method involves randomly selecting a subset of the authentic samples to stand in as authentic test samples. Napier *et al.* advise caution, however, noting that bootstrapping might yield artificially low error rates.[24] They do use this method, however, noting that as long as it is used consistently, they

can compare their results within their own set of experiments. The disadvantage is, therefore, that their results cannot be directly compared with those from experiments not implementing bootstrapping. The practice of bootstrapping is based on the assumption that the user's typing style does not vary much between sessions. This is the very same assumption already used for the premise of the entire typing style based authentication system. Therefore, it would seem to be a safe practice. It is very important, however, that the same sample never simultaneously stands both as a part of the template and as a test sample.

In light of the review of prior work for passphrase authentication as reported in the recent literature, goals of this study encompassed the following:

- Eliminate the requirement and need for imposter samples in the training phase of the classification system.
- Verify and validate the results of Obaidat and Sadoun study with the ART2 neural network.[26]
- Explore and fine-tune values of parameters of the ART2 neural network classifier for the user authentication task.

The ART2 neural network was chosen for the classification task of distinguishing between the authentic and impostor passphrase samples because it is capable of classification using only authentic samples in an unsupervised training mode. A public domain software simulator for the ART2 was utilized in the simulation study.[11]

2. Software-Based Authentication System: Simulation and Testing

In order to test the hypothesis of this study, which states that the ART2 neural network based classifier can perform typing-style based user authentication without impostor samples, a software-based system was developed. This system consists of scripts to acquire typing samples, to organize and manipulate data, and to perform simple tests as well as a software implementation of the ART2 neural network to perform classification tasks. A script produces a graphical user interface (GUI) through which users are prompted to type their names and those of others. The script records the intervals between their keyboard events, the key codes produced, and each type of keyboard event (press and release). A script assembles individual users' data into a data set for each group of users. Another script then analyzes the authentic samples using the ART2 neural network paradigm in order to extract a representative set of authentic samples for each user and to determine a suitable measure of the consistency of the user's samples. These representative sets of samples then become the template for the testing phase. In testing, a script compares impostor samples to the authentic sample base using the ART2 neural network algorithm. The script records individual and group statistics for impostor passes/acceptances.

The overall authentication procedure was divided into two sections: data collection and analysis. The data collection process simply records typing samples from groups of users. The analysis process, on the other hand, has several phases, all of which involve the ART2 neural network.

2.1. *Data acquisition protocol*

Underlying decisions for the data acquisition, content, and format included the following:

- All intervals are captured: no effort is made to determine which keyboard event belongs to which key when logging the sequence of intervals.
- The passphrase is the user's name, including capitalized letters and spaces.
- Samples are collected from participants only once.

A program was written in Tcl/Tk 8.0 to capture time intervals between every key press and release as a user types. Using Tcl/Tk under Microsoft Windows NT, system times can be captured with approximately millisecond resolution, a resolution with which Brown and Rogers were successful.[2] All sample collection was performed on the Windows NT platform. When a prototype of this program was implemented in Java (Sun Java version 1.1), it was only capable of timings on the order of hundredths of a second (10^{-2} second) on the Windows NT platform. The ART2 neural network could not effectively distinguish impostors from authentic users when samples were measured at this low resolution. This reinforced the decision to use an environment that provided millisecond resolution in keyboard event timing, namely Tcl/Tk on Windows NT.

The first interval begins with the first keyboard event and ends with the next keyboard event, whether that event is a release or another press. Each successive interval is the time between the next keyboard event and the previous one. This protocol automatically incorporates the measuring of digraph latencies. A digraph is the overlapping of keystrokes as shown in Fig. 1.

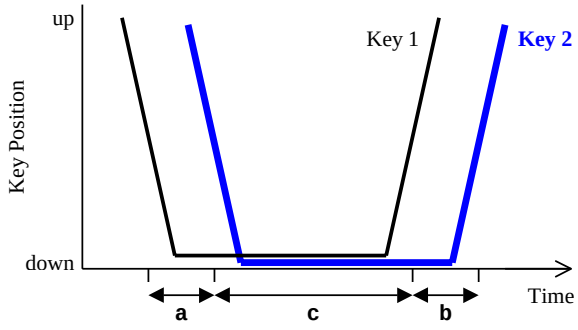

Fig. 1. Diagram of a digraph.

A digraph produces different time interval patterns than "normal" keystrokes. In Fig. 1, **a** is the time between the two successive key presses or the "digraph latency". **b** is the time between the two successive key releases, which is arguably another form of "digraph latency". **c** is the digraph hold time or overlap. Typical keystroke hold times for a given user will be similar to **c**, but due to digraphs, much shorter intervals are produced as in **a** and **b**. This figure only shows one type of digraph. Another example of a digraph would be if **Key 2** were released before **Key 1**. This would be more consistent with a typical capitalization keystroke, whereas the digraph diagrammed here is more indicative of overlapping lowercase keystrokes. Digraph latencies are measured as the time interval between the adjacent key presses and releases (intervals **a** and **b** in Fig. 1).

The software utility to collect user passphrase samples was constructed to deal with one user at a time and deployed to many machines in a computer laboratory so that multiple copies could be used simultaneously. A user first chose his or her name from the list of users in that group. The user then typed his or her name alternated with the names of others in the same test group. The user was prompted to type his or her name as many times as it took to provide one sample for each of the others in the group. The user was forced to pause for two seconds before each prompt.

When the data had been collected from each user, it was moved to a central location with each user's data in a separate directory. A Tcl script merged the data into a set of files for the group. Analysis and testing was performed on this merged data set. The assembled authentic data for a user might have resembled to that of Fig. 2. The impostor data would have been similar except that the key codes would be recorded separately for each attempt.

(a) Keystroke timings (in milliseconds)

```
90 80 11 140 100 561 80 20 60 50 100 130 120 221 260 60 50 221 70 260 60 50 50 70 61
141 70 20 160 120 150 70 51 30 310 90 631 80 210 271 80 70 220 70 131 80 40 40 80 90
100 61 10 190 110 531 70 50 30 70 120 291 80 180 130 70 100 211 90 120 80 20 60 70 81
150 110 10 180 81 200 80 10 70 80 80 261 90 180 210 50 211 240 50 160 70 71 40 120 80
80 100 10 200 110 181 80 10 70 50 80 180 91 200 130 60 90 221 70 130 60 40 50 100 80
100 100 20 131 130 140 90 10 80 40 90 221 80 150 130 70 60 211 90 90 60 50 40 90 60
131 80 30 110 100 140 60 20 50 60 81 250 100 90 100 70 71 130 70 50 80 40 40 80 80
```

(b) Key codes

```
16 68 65 86 73 68 32 16 77 89 69 82 83
```

(c) Event types: (p)ress and (r)elease

```
p p r r p r p r p r p r p r p p r r p r p p r r p r
p p r r p r p p r r p r p r p p r r p r p p r r p r
p p r r p r p p r r p r p r p p r r p r p p r r p r
p p r r p r p p r r p r p r p p r r p r p p r r p r
p p r r p r p p r r p r p r p p r r p r p p r r p r
p p r r p r p p r r p r p r p p r r p r p p r r p r
p p r r p r p p r r p r p r p p r r p r p p r r p r
```

Fig. 2. Snapshot of data acquisition file contents.

2.2. *ART2 neural network classifier*

The Adaptive Resonance Theory neural networks possess the ability to cluster patterns based on their inherent similarities and learn in an unsupervised mode. The ART family of networks boast controlled discovery of clusters and can accommodate new clusters without disturbing previously learned patterns.[13] An ART class neural network automatically generates its own output layer topology and three of four of its parameters can remain constant in the presence of different types of data. The ART family neural networks are capable of reliable processing of complex spatio-temporal data patterns.

All ART neural networks share the same basic core. The core of an ART class neural network consists of a single layer of output nodes (F_2) fed from a single layer of input nodes (F_1). Every input node has a "bottom-up" connection to every output node. Every output node has a "top-down" connection to every input node. In addition, each output node has a "unity delay" connection with unity weight back to its own input and a "unity delay" connection with weight — ε, $\varepsilon \in R^+$, to the input of every other output node. Except for connections between each of the input nodes, the network topology is essentially a complete graph. The ART2 neural network offers capability for normalization and smoothing of continuously valued data through an added layer added to the input section of the ART core topology.

The output of the F_2 layer, $\mathbf{y}$, an $m \times 1$ vector, is initialized by the scalar product similarity measure between the input, $\mathbf{x}$, an $n \times 1$ vector, and the bottom-up weight vector $\mathbf{w}_m$, an $n \times 1$ vector, as in the following equation:

$$y_m^0 = \mathbf{w}_m^t \mathbf{x}, \quad \text{for } m = 1, 2, \ldots, M \tag{1}$$

where $\mathbf{w}_m = [\mathrm{w}_{1m}\mathrm{w}_{2m}\cdots\mathrm{w}_{nm}]^t$ and M is number of nodes in the output layer. The superscript 0 attached to the y_m indicates the 0th iteration for the mth output node. Each further iteration is produced by the following equation:

$$\mathbf{y}^{k+1} = \Gamma[\mathbf{W}\mathbf{y}^k] \tag{2}$$

where $\mathbf{W}$ is the matrix of bottom-up weights on the connections between the input layer and the output layer and Γ is a non-linear diagonal matrix operator.[33] For sufficient recursion index k, the single non-zero output is produced by the jth output layer neuron. This neuron is the "winning" neuron. Its output value is determined by the following equation:

$$y_j^k = \sum_{i=1}^{n} w_{ij} x_i = \max_{m=1,2,\ldots,M} \left(\sum_{i=1}^{n} w_{im} x_i \right). \tag{3}$$

Thus, the weight vector $\mathbf{w}_j$ that best matches the input vector $\mathbf{x}$ is selected for adaptation. "The top-down part of the network checks the similarity of the candidate cluster with the stored cluster reference data and performs the vigilance

test on normalized $\sum_{i=1}^{n} v_{ij}x_i$ entries" as described by the following equation:

$$\frac{1}{\|\mathbf{x}\|}\sum_{i=1}^{n} v_{ij}x_i > \rho \tag{4}$$

where ρ is the vigilance parameter, v_{ij} is the top-down weight from the jth node in F_2 layer to the ith node in F_1 layer, and the norm of the vector $\mathbf{x}$, $\|\mathbf{x}\|$, is defined as follows:

$$\|\mathbf{x}\| = \sum_{i=1}^{n} |x_i| \,. \tag{5}$$

If the vigilance test fails, the output of node j is set to 0, thereby "disabling" this node for the rest of the learning process for the current input. A new winning node is chosen without considering node j and the vigilance test is repeated. When node j passes the vigilance test, the weight matrices are updated as follows:

$$w_{ij}(t+1) = \frac{v_{ij}(t)x_i}{0.5 + \sum_{i=1}^{n} v_{ij}(t)x_i} \tag{6}$$

and

$$v_{ij}(t+1) = x_i v_{ij}(t)$$

where $i = 1, 2, \ldots, M$. The network is originally initialized for n-tuple input vectors as follows:

$$\mathbf{W} = \left[\frac{1}{1+n}\right] \tag{7}$$

$$\mathbf{V} = [1]$$

and

$$0 < \rho < 1 \,.$$

The vigilance parameter, ρ, is the only parameter that must be provided to the network. Practically speaking, the functional range for ρ is approximately $0.70 < \rho < 0.99$. The vigilance parameter determines the level of similarity required in order for the current input vector to belong with the current tentative "winning" cluster.

The recall mechanism of the ART is not used in this study. The clustering is performed anew with each test sample, thus the recall functionality of the ART is not described. The analysis program was written in Tcl 8.0 (without a Tk graphical user interface). The phases of the analysis process are presented next.

2.2.1. *Filtering authentic samples and training the network*

The authentic sample sets likely contain unrepresentative samples. The user might have paused slightly, even imperceptibly, for any of a number of reasons, causing one or more intervals to be disproportionately longer than in other samples. These tainted samples were removed from the sample set using a "preprocessing" technique. Due to the intentional lack of impostor samples, checking for "contaminated"

clusters was impossible. Without any precedent, it was an open question as to how the unrepresentative authentic samples should be sorted out. This problem introduced several questions: how small can a cluster be and still be a representative cluster? What is an acceptable number of representative clusters for a single user? When should pruning of outlier samples stop in favor of keeping a large enough overall set of samples?

These questions were incorporated into the preprocessing procedure as four variables: *minClusterSize*, *stylesMin*, *stylesMax*, and *sampleMin*. The variable *sampleMin* was held constant at a value of 8. Initial tests which allowed a minimum of 6 samples yielded poor results. It is logical that fewer authentic samples would limit the ability of the neural network to distinguish between future authentic samples and impostors. Typing "styles" are represented by clusters of samples, so the terms "cluster" and "style" are used interchangeably. A user would logically have a minimum of one style and a very consistent typist may have exactly one style. Thus, *stylesMin* was held constant at a value of 1. It was uncertain what a safe upper bound on the number of styles should be. Too many styles would increase the impostor pass rate, but too few would increase the false alarm rate. *stylesMax* was therefore given two values, 3 and 4, to see if constraining this parameter had any effect on the outcome. It was also uncertain whether clusters should be restricted to those consisting of more samples in order to ensure the strength of the clusters, or if doing so would only exclude too many representative samples and weaken the overall set of authentic samples. *minClusterSize* was therefore given two values, 2 and 3. All four combinations of these two variables were used and analyzed.

The samples are first clustered by the ART2 neural network starting with a high vigilance parameter (ρ) value of 0.98. The high ρ value tends to allow only nearly identical samples to be clustered together. This results in a large number of small clusters. Outliers are then defined as clusters consisting of fewer than *minClusterSize* samples. If, after the removal of outliers, the number of representative clusters is between *stylesMin* and *stylesMax* (inclusive) and the number of samples is at least *sampleMin*, then the vigilance parameter is assigned to the user and the sample set is pruned accordingly. Otherwise, the vigilance parameter value drops by 0.01 and the cycle repeats, until either the conditions are fulfilled or twenty tries elapse without success. This implies a range for ρ of [0.79, 0.98]. The lower bound on the range is not unreasonable. The ART2 handles vigilance differently from other neural network algorithms. A vigilance value less than approximately 0.7 will have no effect on the algorithm. This means that as the ρ value approaches 0.7, the ability of the network to discriminate between different patterns diminishes asymptotically.[11]

This preprocessing procedure was determined in an attempt to produce a representative set of authentic samples with a high vigilance parameter. An alternative attempt was to start with a midrange value for the vigilance parameter and then to adjust the parameter up or down to fit the nature of each user's samples. This method produced significantly lower values for the vigilance parameter. It is

expected that a range of values for the vigilance parameter would produce acceptable representative sample sets. The importance of choosing the highest value in that range lies in the phase of analysis in which the impostor samples are tested against the filtered authentic samples. The higher the ρ value, the lower the likelihood of impostor samples being clustered with authentic samples.

2.2.2. *Testing the impostor samples*

The impostor samples for each user were tested, one at a time, against that user's set of filtered authentic samples. One of the metrics for gauging the success of this overall approach to authentication was derived directly from the results of this phase of the analysis process. The impostor pass rate (IPR) was the ratio of impostor samples that succeeded in being clustered with authentic samples to the total number of impostor samples tested. This metric is highly correlated with the vigilance parameter. The higher the vigilance, the less likely it is that an impostor sample will be accepted. Of course, the vigilance parameter depends on the nature of the authentic samples as seen in the preprocessing phase. The more consistent a typist the authentic user is, the higher the user's threshold will be, and thus the lower the user's IPR will be.

For each user, the reduced representative authentic samples were copied into a temporary file. The first of that user's impostor samples was then appended to the file. The untrained ART2 neural network was then provided with that file. Usually, a trained network would be reused with each impostor sample, but the penalty for retraining the network for each impostor was negligible. This allowed us to use a "clean" neural network for each impostor sample. The output layer of the ART2 neural network was then analyzed to determine whether the impostor sample was assigned an output node to which any authentic samples had also been assigned. If this was the case, then the impostor sample was said to have "passed". This procedure is repeated for each impostor sample for the user while keeping a count of the impostor passes. Then the user's individual IPR value was calculated. This procedure was then repeated for each user in the group while keeping a count of the total number of passes and the total number of impostor samples. Then the group IPR value was calculated.

Note that it is not necessary for every user to type differently from one another for this authentication scheme to work. Individual styles are brought out when users type familiar phrases, such as one's own name. Two users with similar styles will type their own names fluidly, but will not type another user's name with such alacrity. In addition, before attempting to pose as another person, the intruder has no idea whether practicing that person's passphrase to fluidity will actually result in a success. For all the impostor knows, a particular user may not type fluidly at all.

2.3. *Testing and results*

Upperclassmen of an American university studying engineering disciplines were presumed to have had sufficient exposure to computing environments to have

developed a fluid (or at least an idiosyncratic) typing style. Three groups of 16, 17, and 17 students, respectively, were selected for participation in testing. 7, 14, and 8 students, respectively, responded by providing typing samples of their own names. The format for the names was first name then last name, separated with a space and capitalized as they would be in a formal document. Each participant also provided one impostor sample for each of the other users in the group. These samples were analyzed by the ART2 neural network to determine a representative set of authentic samples for each user. Then the ART2 neural network was used to test the impostor samples for each user against the representative authentic sample set for that user. From these tests, an impostor pass rate (IPR) is calculated for each user and an average IPR is calculated for each group.

Several methods of training using the authentic samples were devised. All methods began with a very high vigilance parameter (ρ) value of 0.98. The authentic samples were clustered by the ART2 neural network to eliminate unrepresentative samples (outliers) from the authentic sample set. The high ρ value at first causes many samples to be labeled as outliers. The test is performed iteratively with successively lower values for ρ until a certain set of conditions are satisfied. The relevant conditions are embodied in three parameters: the number of representative samples (samples remaining after outliers have been eliminated), the number of representative clusters (clusters consisting of representative samples), and the minimum number of samples in a cluster (the size of the cluster is proportional to its "strength"). The most potent effect of these conditions was on the final value of ρ. This value of ρ was assigned to the user along with the corresponding set of representative samples. The user's ρ value was then used when testing impostor samples and thus had a strong effect on the IPR.

The parameters were first allowed to vary in order to explore the impact of each parameter on both the final ρ value and the IPR. Brown and Rogers fixed the minimum number of samples in a cluster at 2.[2] In order to allow for the possibility that larger (i.e. stronger) clusters might decrease IPR, the value was allowed to vary between 2 and 3. The minimum number of representative clusters was fixed at two. To allow a user to have a single cluster would lower the IPR, but would drastically raise the false alarm rate. The maximum number of representative clusters was allowed to vary between 3 and 4. Too many clusters would make the sample set more vulnerable to impostors. The minimum total number of representative samples was fixed at 8 to start with. The summary of the results of these initial tests is presented below in Table 1.

The results indicate that requiring 3 samples in a cluster lowers the final ρ value and significantly increases the IPR. Also, note that when testing with a minimum of 3 samples in a cluster, there was no difference between limiting the number of clusters to 3 or 4. In further testing, the minimum number of samples in a cluster was fixed at 2, thereby eliminating one of the variables.

There was also not a significant difference in results between limiting the number of clusters to 3 or 4 when enforcing only a minimum of two samples in a cluster. The

Table 1. Summary of results for the first round of testing. The minimum number of total samples was fixed at 8.

	Minimum Samples in a Cluster	Final ρ		IPR (%)	
		Max 3 Clusters	Max 4 Clusters	Max 3 Clusters	Max 4 Clusters
Group 1	2	0.921	0.920	9.6	14.3
	3	0.900	0.900	19.1	19.1
Group 2	2	0.934	0.863	13.5	12.4
	3	0.899	0.899	23.0	23.0
Group 3	2	0.943	0.940	4.8	4.9
	3	0.923	0.923	17.9	17.9

Table 2. Summary of results for the second round of testing. The minimum number of total samples was fixed at 6. The minimum number of samples in a cluster was fixed at 2.

	Final ρ		IPR (%)	
	Max 3 Clusters	Max 4 Clusters	Max 3 Clusters	Max 4 Clusters
Group 1	0.903	0.904	19.1	19.1
Group 2	0.903	0.904	24.7	26.8
Group 3	0.926	0.926	18.1	16.5

only exception to this is in Group 1, which appears to be an anomaly because the final ρ value did not change significantly. Unsure of the validity of this observation, another set of tests was performed with the minimum total samples set at 6 instead of 8. The intention was to allow the algorithm more freedom to discriminate between the authentic samples. The results are summarized in Table 2.

The second round of testing did not reveal any relationships between the testing parameter and the results. It also produced far poorer results than in the first round. It did establish, however, the rough correlation between lower ρ value and higher IPR. With the intention of keeping the ρ values high, three methods of training were devised allowing varying degrees of "looseness" in determining representative authentic sample sets within the parameters. These methods are labeled as M-1, M-2, and M-3 in the tables that follow, and are characterized by two features. The first feature is the condition upon which the preprocessing algorithm settles on a final value for ρ. The second feature is the method by which the ρ value is adjusted between iterations of the preprocessing algorithm.

- Method 1 (M-1) was the most "strict". The preprocessing algorithm iterated until both an acceptable number of clusters had been identified and the total number of samples in those clusters was sufficient. It adjusted the value for ρ down by 0.01 if there were too few clusters and up by 0.01 if there were too many

clusters. It also adjusted the value for ρ down by 0.01 if there were an acceptable number of clusters but too few total samples.

- Method 2 (M-2) was less strict. The preprocessing algorithm iterated until either an acceptable number of clusters had been identified or the total number of samples in those clusters was sufficient. It adjusted the value for ρ down by 0.01 if there were too few clusters and up by 0.01 if there were too many clusters. It also adjusted the value for ρ down by 0.01 if there were an acceptable number of clusters but too few total samples.
- Method 3 (M-3) was the "loosest". The preprocessing algorithm iterated until an acceptable number of clusters had been identified. It completely ignored the total number of samples in those clusters. It adjusted the value for ρ down by 0.01 if there were too few clusters and up by 0.01 if there were too many clusters. It did not adjust the value for ρ based on the total number of samples.

The third round of testing performed the preprocessing on all groups using each of the three methods and varying the maximum number of clusters. In all tests, the minimum total number of representative samples was fixed at 8. In tests using method 3, however, the minimum total number of representative samples was not considered. The range for the maximum number of clusters was broadened to allow for the possibility of better results outside of the expected range. The values for this parameter were varied between 2 and 5 clusters. The results are summarized in Table 3.

The results from the third round of testing indicated that method 3, the "loosest" algorithm, produced the lowest average IPR. In the presence of the loose algorithm, the limit on the number of styles had no consequence on the results. In the presence of the "strict" method 1 algorithm, however, the stronger limit of only 2 clusters resulted in a lower IPR in most cases. The average ρ value and IPR

Table 3. Summary of results for the third round of testing. The minimum number of total representative samples was fixed at 8. The minimum number of samples in a cluster was fixed at 2.

	Method	Final ρ				IPR (%)			
		Max 2 Clusters	Max 3 Clusters	Max 4 Clusters	Max 5 Clusters	Max 2 Clusters	Max 3 Clusters	Max 4 Clusters	Max 5 Clusters
Group 1	M-1	0.934	0.923	0.921	0.920	5.9	7.8	11.8	11.8
	M-2	0.929	0.920	0.921	0.920	5.9	11.8	11.8	11.8
	M-3	0.937	0.937	0.937	0.937	5.9	5.9	5.9	5.9
Group 2	M-1	0.940	0.936	0.931	0.931	14.7	14.1	12.5	12.5
	M-2	0.939	0.931	0.931	0.931	13.0	14.1	12.5	12.5
	M-3	0.944	0.944	0.944	0.944	10.9	10.9	10.9	10.9
Group 3	M-1	0.950	0.945	0.940	0.940	0.0	3.4	5.1	5.1
	M-2	0.948	0.940	0.940	0.940	3.4	5.1	5.1	5.1
	M-3	0.955	0.955	0.955	0.955	0.0	0.0	0.0	0.0

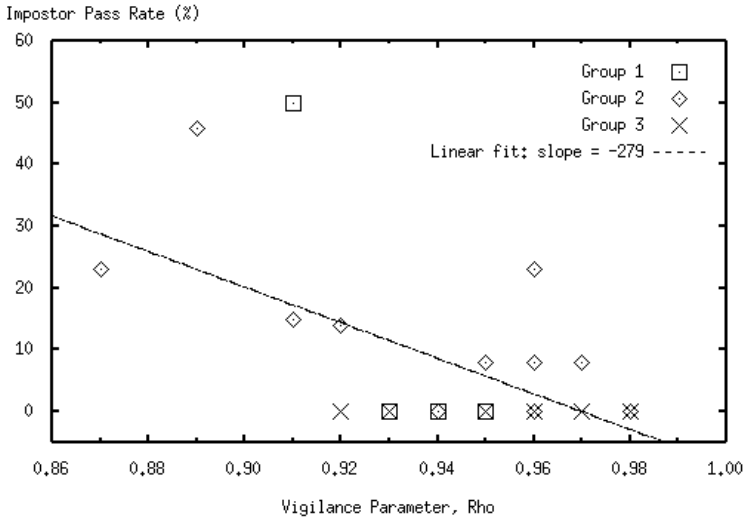

Fig. 3. Impostor Pass Rate (IPR) versus ART2 Vigilance Parameter (Rho).

seem to be strongly correlated in Table 3. As the value for ρ rises, the IPR falls. This relationship is detailed in graphic form in Fig. 3, which shows the IPR versus ρ relationship for all three groups and includes a linear regression fit indicating the negative correlation between IPR and ρ value confirming the observation that IPR falls as ρ rises.

3. Comparative Performance Evaluation

In the application of neural network algorithms to the task of user identification, a few studies stand out in their significance and deserve a detailed investigation and comparison with the methodology proposed in this paper. An earlier study suggested that statistical and pattern recognition techniques were inferior to neural network-based classification techniques for the user authentication task. Specifically, a set of algorithms in the domain of statistics and pattern recognition theory including K-means, cosine measure, minimum distance, Bayes' rule, and the potential function were applied to the task of passphrase authentication. Neural network-based algorithms were found to perform markedly better.[25,26] In light of this finding, comparative performance evaluation of the proposed ART2 authentication algorithm with other significant neural network based paradigms reported in the literature is warranted and will be presented next. The work of Brown and Rogers,[2] Obaidat,[25] and Obaidat and Sadoun[26] are the most prominent neural network based studies to date for the user authentication task.

Brown and Rogers employed a single type of neural classifier, a multi-layer perceptron (MLP) with back-propagation (BP) training, to discriminate between the typing samples of authentic users and those of impostors.[2] Obaidat and Sadoun

employed Fuzzy ARTMAP, radial basis function (RBF), and linear vector quantization (LVQ) neural networks, which reportedly performed superior to other neural and non-neural classification algorithms.[26] They also investigated the effects of variations on the MLP neural network with BP training. Our study focused on the Adaptive Resonance Theory 2 (ART2) neural network for the pragmatic reason that it does not require impostor samples during training while lending itself to online training as well, which offers typing style-based authentication technology a great leap forward toward use in a practical setting. The previously reported combined error rate of zero percent for the fuzzy ARTMAP algorithm also established a strong motivation to explore ART class algorithms further, both to fine tune its settings for applicability in real-world situations, and to validate the results reported by Obaidat and Sadoun.[26]

Apart from the choice of neural network classifier, all three studies report significant differences in data representation as well as information content, acquisition, and pre- and post-processing. A high-level look at these differences provides valuable insight to the effect that reported performance results are only meaningful and valid within the often-unique context in which they were obtained.

Brown and Rogers data was collected at millisecond accuracy. One group included 25 junior and senior business students and the other group 21 senior business students. Fifteen additional students provided impostor samples for the other groups. The represented range of typing skills was wide at a calculated 15 to 40 words per minute. Data was collected at the participants' leisure over a period of several weeks. The data collection utility prompted the users to type their own names and the names of the other participants in the group. The names of others were interspersed between the prompts for the user's own name. The strings averaged about 15.6 characters in length. All keyboard events were recorded and both key hold times and digraph latencies were used to construct the feature vectors.[2] The data collection protocol was similar in the Obaidat studies. Fifteen participants of unreported typing skill provided 225 samples per day for 8 weeks. Then each provided 15 impostor samples for each of the other participants in a single session. The samples were user IDs averaging 7 characters in length. Samples were collected with an accuracy of tenths of a millisecond (0.0001 second).[25,26] Finally, this study recruited three groups of participants numbering 7, 8, and 14. The generally low number of participants in all studies indicates that peculiarities of the population sample might account for a great deal of the relative success or failure of a study.

This study drew its participants from upper class engineering students, presumably with a long exposure to typing tasks. When considering only authentic samples, the mean typing speed was 6.25 keystrokes per second and the standard deviation was 2.01. The slowest speed was 3.37 and the fastest speed was 10.40 keystrokes per second. Typing speed was similar across the three groups (the standard deviation of the means was only 0.25). When considering only impostor samples, the mean

typing speed was 4.05 keystrokes per second and the standard deviation was 0.49. The slowest speed was 3.02 and the fastest speed was 5.11 keystrokes per second. Brown and Rogers drew participants from the nearby business college, and typing skill varied widely. The slowest speed was 1.94 and the fastest speed was 7.74 keystrokes per second, with most falling between 3.65 and 4.70 keystrokes per second. The group of only senior students was generally faster than the group containing both junior and senior students.[2] The Obaidat studies only refer to the participants as volunteers and do not provide any data on typing speed distribution.[25,26] This could explain some of the error rates perceived in both this study and that of Brown and Rogers. The likelihood that this variation in proficiency contributes significantly to variation in results highlights the need for all authors in this subject area to publish data on the skill level of participants.

Brown and Rogers "preprocessed" the raw data to reduce the original set of authentic samples to a set of *representative* authentic samples. A study by Henderson and Mahar measured a statistically significant difference in typing style when users were presented with an increased number and frequency of authentication challenges.[14] In order to eliminate authentic sample sets that are "contaminated", Brown and Rogers implemented three types of preprocessing. Filtering of samples is accomplished using a Kohonen self-organizing map (SOM) neural network.[18] The SOM is presented with a training set consisting of both the authentic and impostor samples for a given user, which groups the samples based on their inherent similarities. These groups, or clusters, may contain both authentic and impostor samples. The first type of filtering eliminates clusters with an impostor-to-authentic sample ratio greater than one-to-six. The second type removes clusters consisting of a single authentic sample (outliers). The third type of filtering consists of applying both the first and the second types, removing both contaminated clusters and singleton outliers.[2]

An unsupervised neural network, ART2 in this case, as a classifier leads to the new situation of having to "preprocess" a group of authentic samples without the availability of impostor samples. The Brown and Rogers study used impostor samples in one of its two preprocessing strategies.[2] By clustering authentic samples with impostor samples, one can look for clusters containing too many impostor samples and eliminate them as "weak". Without impostor samples, one must rely on the inherent consistency of the authentic samples themselves. In an effort to explore this preprocessing procedure further, this study expands the "outlier" elimination strategy by parameterizing the definition of an outlier.

The work performed in this study uses the full name of the user as the identification string, complete with capitalized letters and spaces. This agrees with the methods of Brown and Rogers,[2] but Obaidat[25] and Obaidat and Sadoun[26] employ user IDs, which are approximately half as long and are presumably all lowercase. The shorter length of user IDs provides less information upon which to perform classification. The lack of capitalization and spaces, however, could decrease overall

variation between samples, especially within the authentic sample set, which could tend to increase the resistance of the system to impostor attacks.

The Obaidat[25] and Obaidat and Sadoun[26] data collection implies the procedure known as "bootstrapping", but it is not explicit about its use of authentic samples. Brown and Rogers explicitly collect two sets of authentic data, several weeks apart, avoiding the need for bootstrapping.[2] This allows them to perform a more realistic assessment of the false alarm rate, which could partially explain the higher error rate encountered.

In the end, the Brown and Rogers study was successful in demonstrating that a user's name is a sufficiently long text on which to base this brand of biometric security.[2] Nevertheless, it was hampered by the MLP's training requirement of roughly twice as many impostor samples as authentic samples. Access to true impostor samples in large quantities is most likely not a realistic assumption for the design of an authentication algorithm.

Both Obaidat studies essentially reached the same conclusions: key hold time is more important than inter-key time, and using both intervals provides the best results. In testing the hypothesis, both studies examined multiple statistical and neural network algorithms, leading to the following ancillary conclusions: neural networks are better-suited to this task than are statistical methods, and certain neural network algorithms excel at the task while others do not. When both interval types were considered, the authors reported that three neural network algorithms achieved 100% accuracy: Fuzzy ARTMAP, radial basis function (RBF), and learning vector quantization (LVQ) networks.[25,26]

Perhaps the most important conclusion regarding the ART2 classifier proposed in this study is that the lack of impostor training samples is not an impediment for the successful design of a passphrase-based authentication algorithm. Furthermore, the ART2 neural network classifier can also be used in an online incremental training mode, characteristic of the classifier, in order to test unknown samples.

In light of numerous significant differences in data collection, sample size, and preprocessing techniques, and uncertainty about the algorithmic settings as well as training and testing procedures among the three studies reported and subsequently compared in this section, it is necessary to state that a meaningful comparison is most likely not feasible. Obaidat appears to have achieved the best classifier performance for the typing style-based authentication task reported in the literature, although the results of our study failed to validate their favorable findings.

4. Conclusions

The goal of this study was to explore the feasibility of training an unsupervised and self-organizing artificial neural network to authenticate users based on typing style without the use of imposter samples. Prior methods relied on the collection of "impostor" samples to be used in training the neural network. In the target environments for this type of biometric authentication technology, foreknowledge

of impostor behavior is unrealistic and a dependence on simulated impostor data should be seen as a weakness. This study demonstrated a training method that does not have this impractical requirement and thus avoids a weak reliance on simulated impostor samples. Results indicated that the ART2 neural network classifier can achieve imposter pass rates on the order of 10% and therefore, bring typing style based password authentication closer to the realm of deployment in real-world target environments. Findings of this study predict that the no-impostor approach has the potential for practical deployment in this type of biometric authentication system.

Acknowledgments

Authors gratefully acknowledge Dr. Hannah Chen, Professor, Computer Science Department, University of Alabama, for her suggestions with the ART2 neural network, the Institute for Computer Science and Social Studies' Telematik Department at the Albert-Ludwigs University of Freiburg Germany, for the conducive environment to produce manuscript revisions, The United States National Science Foundation for grant support under the Research Experiences for Undergraduates Program, Grant No. CDA-9619680, the Honors Program at the University of Toledo, Ohio, for encouraging this work as an Honors thesis, and reviewers of this manuscript for the valuable feedback provided.

References

1. J. Bechtel, Authentication of users using typing styles through an ART2 neural network, Unpublished Honors Thesis, Electrical Engineering and Computer Science Department, The University of Toledo, Toledo, Ohio, USA, August 2000.
2. M. Brown and S. J. Rogers, User identification via keystroke characteristics of typed names using neural networks, *Int. J. Man-Machine Stud.* **39** (1993) 999–1014.
3. M. Brown and J. Rogers, A method and apparatus for verification of a computer user's identification based on keystroke characteristics, Patent Number 5,557,686, U.S. Patent and Trademark Office, Washington D.C. (1996).
4. O. Coltell, G. Torres and J. M. Badía, Biometric identification system based in keyboard filtering, *33rd Ann. IEEE Int. Carnahan Conf. Security Technol.*, October 1999, 203–209.
5. P. S. Dowland, H. Singh and S. M. Furnell, A preliminary investigation of user authentication using continuous keystroke analysis, *Proc. IFIP 8th Ann. Working Conf. Info. Security Manag. Small Syst. Security*, September 2001.
6. W. G. de Ru and Jan H. P. Eloff, Enhanced password authentication through fuzzy logic, *IEEE Intell. Syst. Their Appl.* **12** (November/December 1997) 38–45.
7. L. V. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications* (Prentice Hall, Englewood Cliffs, NJ, 1994) 222–288.
8. G. Forsen, M. Nelson and R. Staron, Personal attributes authentication techniques, Rome Air Development Center, Report RADC-TR-77-1033, ed. A. F. B. Griffis, RADC, New York, 1977.
9. R. Gaines, W. Lisowski, S. Press and N. Shapiro, Authentication by keystroke timing:

Some preliminary results, *Rand Report R-256-NSF*, Rand Corporation, Santa Monica, 1980.
10. J. Garcia, Personal identification apparatus, Technical Report Patent Number 4,621,334, US Patent and Trademark Office, Washington, D.C., 1986.
11. P. Gaudiano, Source code for ART2 neural network simulator, Boston University, Department of Cognitive and Neural Systems, March 1990.
12. D. R. Gentner, Keystroke timing in transcription typing, *Cognitive Aspects of Skilled Typewriting*, ed. W. E. Cooper (Springer, Berlin, 1983) 95–120.
13. S. Grossberg and G. Carpenter, ART 2: Self-organization of stable category recognition codes for analog input patterns, *Appl. Opt.* **26** (1987) 4919–4930.
14. R. Henderson and D. Mahar, Electronic monitoring systems: An examination of physiological activity and task performance within a simulated keystroke security and electronic performance monitoring system, *Int. J. Human-Computer Stud.* **47** (1998) 143–157.
15. R. Joyce and G. Gupta, Identity authentication based on keystroke latencies, *Commun. ACM* **33** (February 1990) 168–176.
16. A. A. Kaji and M. Analoui, Computer network-access security using keystroke dynamics, a covariance-covariance approach, *Proc. Int. Symp. Telecommun.*, Tehran, Iran (September 1–3, 2001).
17. H.-J. Kim, Biometrics, Is it a viable proposition for identity authentication and access control?, *Comput. Security* **14** (1995) 205–214.
18. T. Kohonen, *Self-Organization and Associative Memory*, 3rd edn. (Springer, Berlin, 1989).
19. J. Leggett and G. Williams, Verifying identity via keystroke characteristics, *Int. J. Man-Machine Stud.* **28** (January 1988) 67–76.
20. J. Leggett, G. Williams and M. Usnick, Dynamic identity verification via keystroke characteristics, *Int. J. Man-Machine Stud.* **35** (1991) 859–870.
21. D.-T. Lin, Computer-access authentication with neural network based keystroke identity verification, *Proc. IEEE Int. Conf. Neural Networks* **1** (June 1997) 174–178.
22. J. Long, I. Nimmo-Smith and A. Whitefield, Skilled typing: A characterization based on the distribution of times between responses, *Cognitive Aspects of Skilled Typewriting*, ed. W. E. Cooper (Springer, Berlin, 1983) 145–196.
23. D. Mahar, R. Napier, M. Wagner, W. Laverty, R. D. Henderson and M. Hiron, Optimizing digraph-latency based biometric typist verification systems: Inter- and intratypist differences in digraph latency distributions, *Int. J. Human-Computer Stud.* **43** (1995) 579–592.
24. R. Napier, W. Laverty, D. Mahar, R. D. Henderson, M. Hiron and M. Wagner, Keyboard user verification: Toward an accurate, efficient, and ecologically valid algorithm, *Int. J. Human-Computer Stud.* **43** (1995) 213–222.
25. M. S. Obaidat, A verification methodology for computer systems users, *Proc. 1995 ACM Symp. Appl. Comput.* (1995) 258–262.
26. M. S. Obaidat and B. Sadoun, Verification of computer users using keystroke dynamics, *IEEE Trans. Syst. Man Cybernet.: Cybernet., Part B* **27** (April 1997).
27. D. J. Ostry, Determinants of interkey times in typing, *Cognitive Aspects of Skilled Typewriting*, ed. W. E. Cooper (Springer, Berlin, 1983) 225–246.
28. A. Peacock, Learning user keystroke latency patterns, Brigham Young University, Performance Evaluation Laboratory, http://pel.cs.byu.edu/∼alen/personal/CourseWork/cs572/, Winter 2000.
29. S. J. Shepherd, Continuous authentication by analysis of keyboard typing characteristics, *Eur. Convention Security Detection*, Conference Publication No. 408 (IEE, 1995).

30. D. M. Skapura, *Building Neural Networks* (ACM Press, New York, 1996) 46–52.
31. D. Umphress and G. Williams, Identity verification through keyboard characteristics, *Int. J. Man-Machine Stud.* **23** (1985) 263–273.
32. J. R. Young and R. W. Hammond, Method and apparatus for verifying an individual's identity, Patent Number 4,805,222, US Patent and Trademark Office, Washington D.C. (1989).
33. J. M. Zurada, *Introduction to Artificial Neural Systems* (PWS Publishing Company, 1995) 394–395 and 432–437.