Identity Detection of Typist relying on Image Processing Techniques

Sherif. T. Amin¹, Magdy. A. Saeb², Marghny. H. Mohammed³
¹Student, Faculty of Science, The University of Assyut, Assyut, Egypt
²Professor, Computer Engineering Department, Arab Academy for Science, Technology & Maritime Transport, Alexandria, Egypt
³Assistant Professor, Faculty of Computer and Information Sciences, The University of Assyut
sherif7bacha@yahoo.com, mail@magdysaeb.net, marghny@aun.edu.eg

Abstract
Global access to information and resources is becoming an essential part of nearly every aspect of our lives. Unfortunately, with this global network access comes increased chances of malicious attack and intrusion. In an effort to confront the new threats unveiled by the networking revolution of the past few years reliable, and rapid means for automatically recognizing the identity of individuals are now being sought. In this paper we consider the problem of identifying a user typing on a computer keyboard, through identification of his behavioral typing patterns and the time series consisting of keyboard events. A graphical representation of the user typing behavior in which frequency domain filtering operations are developed to robustly extract useful high-level information from the user's behavioral images, then using cross-correlation as measure of similarity we can accurately authenticate users. Our solution is based on simple template matching methodology. Application of our results could be a second-layer behaviometric security system continually testing the current user without interfering with this user’s work while attempting to identify masquerading users. We study the effectiveness of our method over a real dataset consisting of 17 users and 23 attackers.

Keywords: Behaviometrics, Image Processing, Pattern Matching.

1. Introduction
To protect information systems from unauthorized use, administrators depend on security technologies such as firewalls, network, host-based intrusion detection systems, and strong authentication protocols. If an intruder can gain access to a legitimate user account, however, these state-of-the-art security technologies are rendered useless. For instance, once a user has his password compromised by a cracker, the cracker can then utilize all the privileges available to the legitimate user without being detected due to the trust placed in the compromised account. Similarly, a malicious insider can choose to use their privileges to perform illegal actions. These examples depict the canonical masquerade attack, when an attacker masquerades as a legitimate user of the system to perform unauthorized and malicious actions without being subjected to the scrutiny of traditional security technologies. Clearly, such attacks pose a severe threat, and their detection often occurs long after the damage is done. The key, therefore, is to develop techniques to identify typist based on their typing behaviour and then differentiate between them during their usages of the information system. Of course, this is quite difficult in practice, as legitimate daily activity can simply become malicious based on its context. In fact, there have been several experiments at creating algorithms for detecting these attacks, and yet a level of accuracy required for practical deployment has not been achieved [1, 2, 3, 4]. In this paper, we leverage the pattern matching abilities of template matching algorithms found in the field of digital image processing to identify typist typing behavior within sequences of information system audit data (e.g., key stroke entries, and time delay between key strokes). Template matching algorithms are used in the field of image processing to detect areas of similarity between two digital data images. The ability of template matching algorithms to discover areas of similarity can be used to identify typist when using key board based on their behaviometric records. To do so, we create a signature of the normal behavior for a given user by collecting sequences of audit data that are known to be created from legitimate daily use of the information system, known as the user signature. This user signature can then be aligned with audit data collected from monitored sessions to discover areas of similarity between the two. Areas that do not align properly can be assumed to be anomalous, and the presence of many of these anomalous areas is a strong indicator of misidentification of typist. There are many problems that are inherent to the task of detecting masquerade attacks. First, the usage patterns of legitimate users can be expected to change, perhaps due to new projects or typing language. The use of static user signatures is therefore prone to label legitimate variations as attacks. By using the template matching algorithm’s
ability to detect areas of similarity, we are able to dynamically update the user’s signature as new user behaviour is encountered, thereby avoiding false detection of masquerade attacks. Second, by selectively performing alignments only on the portions of the user signature that have the same key strokes triplet sequence, we significantly reduce the number of computations required in practice with almost no loss of accuracy in detecting typist identity. Our method which takes the template matching algorithm as its alignment tool, along with our scoring scheme, were tested using real typing Arabic essays from 17 users in which each user typed two essays one with 5000 characters and another one with 2500 characters, and 23 users presumed attackers where some of them typed only the 2500 characters essays, and others typed incomplete 2500 characters and 5000 characters essays. The following flowchart depicts the procedures described in sections two, three, and four.

Results of the evaluation show that our system provides good accuracy. These results provide encouragement for the practical use of our template matching technique, and advance the state of the art in Arabic typist detection. The paper is organized into ten sections first we begin by describing the collected data, the collection methodology, and the data preparation process (section 2). We then provide a description of the graphical methodology used to detect our typist (section 3). We continue by describing our proposed graphical solution, the image processing techniques used as a pre-processing step before using cross-correlation as the typist identification process (section 3 and 4). The used testing framework, the performance evaluation measures used in the experimentations, the experimental results, and performance statistics are all respectively depicted in sections (5, 6, 7, 8, and 9). We conclude the paper by drawing attention to the reason behind coding the typist behavioural data in a 2D graphical representation, reviewing the intrinsic limitations of the current modelling methodology and by proposing a few suggestions for future work (section 10).

2. Audit Data pre-processing

The dataset used in our sequence alignment method were collected from 40 individuals on a period of several days and sometimes the same person had to complete one of his personal essays on two different PC’s due to unavailability of the one previously available. In fact, all previous masquerade detection techniques were evaluated, at least in part, using publicly available dataset on the internet, making comparison among masquerade detection algorithms straightforward, and not using Arabic language as the typing dataset. Our dataset was created by recording users’ key strokes as they were provided to the Microsoft Word program. These key strokes were recorded via a python program utility including all the time delays between each key stroke and the next. Key strokes were recorded for forty distinct users, of which all of them were chosen as the users that would make up the dataset. For each of these forty users in the dataset, an Arabic essay, randomly selected from the Internet with 5,000 characters, shown in figure 1, were given to them and required to be typed at their own pace and time leisure. We will call this key strokes sequence the user’s signatures, or training set. An additional 2,500 characters essay, shown in Figure 2, was also recorded from each user in the dataset to make up the set of key strokes that are to be tested for identification. Only seventeen where able to finish writing both essays, the rest of the forty either wrote one essay or didn’t complete any one of them. Some of the test data were taken from both the incomplete 2500, and 5000 characters essays. We can call this set the test data. Table 1 shows seventeen users out of the 40 and the number of keystrokes they typed in order to properly present both his/her signature and test essays.

Figure 1: 5,000 characters signature essay
The key strokes sequences, in both the user's signature and the test data, have been mined through text mining techniques in order to arrange them in the form of unique triplets' sequences, where the triplets' boundaries form the consecutive key strokes sequence occurring as the result of the user typing. The integer in the middle represents the average time for all the same consecutive key strokes sequence available in the essay. For example, if a user types the word banana, this word will be dissected into:

1. B 328.0 A, where 328.0 represents the time difference in nanoseconds between when the user typed b and a.
2. A 772.0 N, where 772.0 represents the time difference in nanoseconds between when the user typed a and n, the first time.
3. N 655.0 A, where 655.0 represents the time difference in nanoseconds between when the user typed n and a, the second time.

This process of key strokes groupings and averaging is the result of text mining each user audit sequence. This triplet's formation process depicts the dataset of forty different users, where each user has a signature made of number of triplets, depending on his behavior and typing skills. We have chosen 900 nanosecond time difference between keystrokes as a ceil, in which we presume that the user changed his psychological mode when he started typing, like for example stopping typing, talking to a person, or day dreaming, so any time difference above 900 nanoseconds will remain 900. For example, competent typists usually type their essay of 5,000 characters in less than 5,500 key strokes and with average time difference below 750, whereas the incompetent ones exceed the 6,500 key strokes, and with a time difference exceeding 750 nanoseconds for the majority of their keystrokes. Figure 3 shows the test essay after the transformation process. Our task, therefore, is to align the triplets' sequence of the user signature in the 5,000-key strokes essay; with triplets sequences generated by the user from 2,500-key strokes test essay, and then determine if the score resulting from the alignment is indicative of the typist identity. The only information about the identity of the typist within the dataset is discovered by searching for triplets that represent typist personal behavior within both the typist signature and test that is not available within other typist triplet's sequences, and then using the sequence alignment algorithm for triplet's comparison.

Figure 3: Test essay after triplets' formation.

### Table 1: Number of users keystrokes in Signature and test data.

<table>
<thead>
<tr>
<th>User</th>
<th># of Keystrokes in Signature (SDU Chr.)</th>
<th># of Keystrokes in Test (SDU Chr.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>5950</td>
<td>7008</td>
</tr>
<tr>
<td>U2</td>
<td>3502</td>
<td>5950</td>
</tr>
<tr>
<td>U3</td>
<td>2104</td>
<td>6499</td>
</tr>
<tr>
<td>U4</td>
<td>3736</td>
<td>7436</td>
</tr>
<tr>
<td>U5</td>
<td>5207</td>
<td>6986</td>
</tr>
<tr>
<td>U6</td>
<td>5207</td>
<td>6530</td>
</tr>
<tr>
<td>U7</td>
<td>4755</td>
<td>9825</td>
</tr>
<tr>
<td>U8</td>
<td>2372</td>
<td>6596</td>
</tr>
<tr>
<td>U9</td>
<td>2946</td>
<td>6888</td>
</tr>
<tr>
<td>U10</td>
<td>3278</td>
<td>6125</td>
</tr>
<tr>
<td>U11</td>
<td>3543</td>
<td>7627</td>
</tr>
<tr>
<td>U12</td>
<td>5214</td>
<td>6506</td>
</tr>
<tr>
<td>U13</td>
<td>4504</td>
<td>7140</td>
</tr>
<tr>
<td>U14</td>
<td>3912</td>
<td>6096</td>
</tr>
<tr>
<td>U15</td>
<td>3406</td>
<td>6302</td>
</tr>
<tr>
<td>U16</td>
<td>3569</td>
<td>5777</td>
</tr>
<tr>
<td>U17</td>
<td>5607</td>
<td>6296</td>
</tr>
</tbody>
</table>

The same process is repeated with step 3 and 5 and results on the new triplet N 655.0 A.
figure 4 shown below which details the signature and test sets of a user, a black background indicates absence of used key buttons; a dim pixel indicates the usage of the two key buttons that intersect at this pixel with a low average time delay between both of them, the pixel brightness changes according to the average time delay, brighter means higher average time delay.

4. Image Processing
The goal of image processing in this section is to robustly extract useful, high-level information from all users’ behavioral images by filtering the frequency domain via Fourier transform, then use cross-correlation as a measure of similarity between all users’ processed behavioral images, a cross-correlation value of 1 indicates perfect similarity between two images. Two methods were used for frequency domain filtering as a high-level information extraction step:

1. Obtaining frequency domain filters from the Sobel spatial filter (see figure 5) that enhances vertical edges.
2. Sharpening the image by using High Pass Gaussian filter (see figure 6) that attenuates the low frequencies and leaves the high frequencies of the Fourier transform relatively unchanged.

In both methods Discrete Fourier Transform filtering was used, this filtering procedure is summarized in figure 7.

Figure 5, shows the Sobel mask used for enhancement of vertical edges (a), absolute value of the frequency domain filter corresponding to the vertical Sobel mask after processing with function that rearranges the outputs of Fast Fourier Transform by moving the zero-frequency component to the center of the array. It is useful for visualizing a Fourier transform with the zero-frequency component to the center of the array. (b), image (c) is the same filter (b) shown as image.

Figure 6 Frequency Domain Filtering Operations

Figure 7: A user’s original behavioral image (a); Result of filtering the same image with a vertical Sobel mask (b); The result of applying a Gaussian high pass filter on the original image.

5. Testing Framework
The following is an overview of our behaviometric testing methodology. A behaviometric signature is any form of behaviometric identifying data (e.g., a still typing behavior image or template of that information).

1. Two sets of behaviometric signatures were assembled: a target and query set. The target set contains the set of signatures that are known to the system (i.e., the Behaviometric database). The query set contains signatures of subjects that are to be compared against the target set. The intersection of these two sets results in a similarity score that is a measure of how similar the query is to the signature. Some of the same subjects are in both sets, 17 subjects out of 40, separate instances of behaviometric signatures for these 17 subjects have been used.

2. For each pair of query and target signatures a match-score has been obtained and stored in a matrix, called a similarity matrix, whose size is query set size by target set size (17 x 74). The match-score is a measure of how similar two behaviometric signatures are. The match-score represents, in case of image matching the correlation score.

3. Gallery and probe subsets have been extracted from the target/query similarity matrix, respectively, to perform “virtual” experiments on a subset of the population. A gallery is any arbitrary subset of the target set. A probe is any arbitrary subset of the query set.

4. Performance statistics for verification are computed from the genuine and imposter scores. Genuine scores are those that result from comparing elements in the target and query sets of the same subject. Imposter scores are those resulting from comparisons of different subjects. Each maximum score in the similarity matrix has been selected as a starting threshold the false-accept rate (FAR) and false-reject rate (FRR) were computed at lesser positions of the starting threshold ( -0.005, -0.01, -0.05, -0.1, -0.5, -1) by selecting those imposter scores and genuine scores, respectively, on the wrong side of this threshold and dividing by the total number of scores used in the test. A mapping table of the threshold values and the corresponding error rates (FAR and FRR) are stored. The complement of the FRR (1 – FRR) is the genuine accept-rate (GAR). The GAR and the FAR are plotted against each other to yield a Receiver Operating Characteristic (ROC) curve, a common system performance measure.

6. False-Accept Rate (FAR)
The FAR is defined as the probability that a user making a false claim about his/her identity will be verified as that false identity. For example, if U2 types U1’ user ID into the biometric login for U1’ PC, U2 has just made a false claim that he is U1. U2 presents his biometric measurement for verification. If the biometric system matches U2 to U1, then there is a false acceptance. This could happen because the matching threshold is set too high, or it could be that U2's biometric feature is very similar to U1’ Either way, a false acceptance has occurred.
The Simple Math

When the FAR is calculated by a biometrics vendor, it is generally very straightforward. Using our example, it is equal to the number of times that U2 has successfully authenticated as U1 divided by his total number of attempts. In this case, U1 is referred to as the "MatchUser" and U2 as the "NonMatchUser." The simple math formula for this looks like the following, where n represents a number uniquely to identify each user:

\[ FAR = \frac{\text{Number of NonMatchUser'S Successful Authentications}}{\text{Number of NonMatchUser's Attempts to Successfully Authenticate}} \]

This gives us the basis for U2 and U1. What if we have another user, David? We could say that U2 and U1 are representative of our user population and just assume that the FAR will be the same for David. Statistically, the more times something is done, the greater the confidence in the result. Thus, to ensure a high probability that the FAR we calculate is statistically significant, we would need to do this for every combination of users we have. We would need to take all the calculated FARs for each user's attempt to falsely authenticate as another, sum them up, and divide by the total number of users. For example, we could take the above formulas and do them over again for each user. We would eventually get something that looks like the following:

\[ FAR(\text{U1}) = \frac{\text{Number of NonMatchUser's Successful Authentications}}{\text{Number of NonMatchUser's Attempts to Successfully Authenticate}} \]

This gives us the basis for U1. What if we have another user, David? We could say that U1 is representative of our user population and just assume that the FAR will be the same for David. Statistically, the more times something is done, the greater the confidence in the result. Thus, to ensure a high probability that the FAR we calculate is statistically significant, we would need to do this for every combination of users we have. We would need to take all the calculated FARs for each user's attempt to authenticate as himself/herself, sum them up, and divide by the total number of users. The result is the mean (average) FAR for all users of a system. For example, we would take the above formulas and compute them for each user. We would eventually get something that looks like the following:

\[ FAR = \frac{1}{N} \sum_{n=1}^{N} FAR(n) \]

Why Is This Important?

The importance of the FAR is the strength of the matching algorithm. The stronger the algorithm, the less likely that a false authentication will happen. U2 has a greater chance of falsely authenticating as U1 at 1:500 than he does at 1:10,000. An example of this would be playing a ring-toss game. In this game, the object is to throw a ring on to a particular peg. The ring represents U2's false authentication attempt. The number of pegs represents the strength of the biometric algorithm. The game board itself represents U1's biometric enrollment. In the first case, there are 500 pegs on which to throw the ring. The peg that needs to be ringed for a winner is not marked. Thus, U2 has a 1 in 500 chance of hitting the right peg. Now, if U2 is playing the same game, but this time there are 10,000 pegs in the same area, U2 now has a 1 in 10,000 chance of hitting the right peg.

Carrying this example further, U1 now needs to authenticate. He knows the layout of the board and what ring to toss his peg onto. He knows this because he is really who he says he is. In the first game, he is faced with 500 pegs. U1 knows what peg to toss his ring onto in order to get authenticated. There is a chance he could miss, which is very low. If at any time U1 does not hit his peg, then he is falsely rejected. That is to say, he has not been authenticated as himself, even though he is U1.

7. False-Reject Rate (FRR)

The FRR is defined as the probability that a user making a true claim about his/her identity will be rejected as him/herself. For example, if U1 types his correct user ID into the biometric login for his PC, U1 has just made a true claim that he is U1. U1 presents his biometric measurement for verification. If the biometric system does not match U1 to U1, then there is a false rejection. This could happen because the matching threshold is set too low, or U1's presented biometric feature is not close enough to the biometric template. Either way, a false rejection has occurred.

The Simple Math

When the FRR is calculated by a biometrics vendor, it is generally very straightforward. Again, using our example, it is equal to the number of times that U1 unsuccessfully authenticated as U1 divided by his total number of attempts. In this case, U1 is referred to as the "MatchUser":

\[ FAR = \frac{\text{Number of NonMatchUser'S Successful Authentications}}{\text{Number of NonMatchUser's Attempts to Successfully Authenticate}} \]

This gives us the basis for U1. What if we have another user, U2? We could say that U2 is representative of our user population and just assume that the FRR will be the same for U2. Statistically, the more times something is done, the greater the confidence in the result. Thus, if we want to ensure a high probability that the FRR we calculate is statistically significant, we would need to do this for every user. We would then need to take all the calculated FRRs for each user's attempt to authenticate as himself/herself, sum them up, and divide by the total number of users. The result is the mean (average) FRR for all users of a system. For example, we would take the above formulas and compute them for each user. We would eventually get something that looks like the following:

\[ FRR = \frac{1}{N} \sum_{n=1}^{N} FRR(n) \]

Why Is This Important?

The strength of the FRR is the robustness of the algorithm. The more accurate the matching algorithm, the less likely a false rejection will happen. Chris has a lower chance of being falsely rejected as himself at 1:500 than he does at 1:10,000. Again, let's use the example of playing a ring-toss game. In this game, the ring
represents Chris’ authentication attempt. The distance between the pegs represents the robustness of the biometric algorithm. The gameboard itself represents Chris’ biometric enrollment. In the first case, there are 500 pegs on which to throw the ring. The peg that needs to be ringed for a winner is known. Thus, Chris has a 1 in 500 chance of hitting the right peg. Chris also has less of a chance of hitting the wrong peg since there is generous spacing between the pegs. Now, if Chris plays the same game, but this time there are 10,000 pegs in the same area, he has a 1 in 10,000 chance of hitting the right peg. There is now also less spacing between pegs, as the playing area is the same size as it was for 500 pegs. So, Chris has a greater chance of landing on the wrong peg because of the pegs’ relative proximity to each other.

8. Experimental Results

Table 2 (a) specifies the results obtained from cross-correlating test images (rows) against the signature images (columns) for the forty different users, all images processed by obtaining the frequency domain filter from the Sobel spatial filter (first method). To maximize the utility of the data set we presumed each user as an attacker to other one. As we can see from the cross-correlation results (results with green background mean successful matching, results with red background mean unsuccessful matching), as we can see we have seven impostors whom their test data gave better results with the users genuine signature data than the genuine users test data ("red background") (U3, U19, U23, U25, U27, U37, and U39), U39 had a mismatch with both U9 and U17. Table 2 (b) shows the users that gave better cross-correlation results, their cross-correlation scores, and the absolute difference between their scores and scores obtained from cross correlating the user’s test data with their own signature data.

Table 3 (a) specifies the results obtained from performing the same previous experiment using a Gaussian high pass filter instead of the Sobel spatial filter used previously for obtaining the frequency domain filter. As we can see from the cross-correlation results, we have only five imposters whose test data gave better results with the users genuine signature data (U3, U11, U19, and U26), U3 had a mismatch with both U5 and U10. Table 3 (b) shows the users that gave better cross-correlation results, their cross-correlation scores, and the absolute difference between their scores and scores obtained from cross correlating the user’s test data with their own signature data.

9. Performance Statistics

Step 4 of the testing methodology, computes the Receiver Operating Characteristics (ROC) curve for our study. This is a method of showing the performance of the biometric system over a range of decision criteria usually shown as a graph that relates the False Reject Rate to the False Accept Rate as the decision threshold varies. Figure 8 shows a ROC curve for the simple maximum rule, the starting threshold for each user is the maximum score represented in the matrix, the False Accept Rate (FAR) and the False Reject Rate (FRR) were both computed by lowering the threshold at different intervals of the maximum score shows in the matrix (-0.005, -0.01, -0.05, -0.1, -0.5, -1). Clearly the use of these fusion threshold techniques enhances the performance significantly.
For example, at a FAR of 0.1% the simple maximum rule has a GAR of 99.76%, which is considerably good. Figure 9 and 10 shows the related FAR and FRR curves respectively.

Figure 8: Receiver Operating Characteristics (ROC curve)

Figure 9: False Acceptance Rate (FAR curve)

Figure 10: False Reject Rate (FRR curve)

10. Conclusion and Future Work

We have introduced an approach to modeling keystroke dynamics of users based on using digital graphical representation of the users keystrokes events, and statistics collected from an individual user. We use template matching in the context of digital image processing, where particular values for the digitally augmented users behavior patterns are used in user identification. As a result of this, our graphical model is capable of retaining more robust behavioral and statistical patterns, achieving performance results approaching 93% true positive (TP) (percentage of masquerading identity caught) in general cases and with 0.7% false positive (FP) (percentage of true identity that are mislabeled as masquerading).

The arguments behind adopting 2-D image pattern analysis to identify the typist although the typing pattern itself is available are two-folds:

- Image analysis is an evolving technique and coding the typist behavior into a 2-D image which captures both horizontal phenomena, such as harmonics, as well as vertical phenomena, such as time delay will greatly allow a more powerful approach to model the problem.
- The 2-D behaviometric image can be analyzed at constant patch “frame-rates”, so it is easier to manipulate than segment-based or boundary-based approaches which have to deal with variable time-length feature vectors.

Other image matching methods such as wavelets and Gabor have been used without significant results; similarities between typing essays of the same typist are quite obvious in the spectral domain.

Our method can be potentially improved in several ways. First, it may be interesting to investigate whether taking relative time-differentials (rather than absolute time-differentials) can improve performance, perhaps by a reduction of the variance caused by the variability in typing speeds of users. This direction is particularly promising when considering the successful technique of [5], which achieved impressive performance on a fixed text by ignoring absolute differential times (but utilizing the relative sizes of trigraph times). Second, applying wavelet-based detection algorithms developed for detection of small targets which can be considered as behavioural patterns in our case.

Third this paper doesn’t address the problem of user’s similarity and/or affinity, in the future such problem of distinguishing between similar users should be considered significantly.

While our results are notable, they can only be viewed as a proof-of-concept due to the limited sample size, and the use of a two sessions for data acquisition. Finally, an advantage of our techniques is that they are not specifically targeted to the keyboard, and can be easily extended to other devices.

Many Artificial Neural Systems (ANS) architectures, such as backpropagation, adaptive resonance, and others are applicable to the recognition of spatial information patterns: a two-dimensional, bit-mapped image of a behaviometric data, for example. In the future, we should describe ANS architectures that can deal directly with both the spatial and the temporal aspects of behaviometric input signals. These networks encode information relating to the time correlation of spatial patterns, as well as the spatial pattern information itself. Defining a spatiotemporal pattern (STP) as time-correlated sequence of spatial patterns for typist identification would be a considered as a good idea for future work.
11. References


Marghny H. Mohamed, Dept. of Computer Science, Faculty of Computers and Information Science, Asyut University, Asyut, Egypt., date of birth: June 1965, received the PhD degree in computer science from the University of Kyushu, Japan, in 2001, and the MS from Asyut university in computer science, in 1993 and BS degrees in Mathematics from Asyut University, Egypt, in 1988. He is an assistant professor in the Department of Computer Science, University of Assiut. He has many publications which in the fields of Data Mining, Text Mining, Information Retrieval, Web Mining, Machine Learning, Natural Language Processing, Pattern Recognition, Neural Networks, Evolutionary Computation, Fuzzy Systems. Dr. Marghny is a member of the Egyptian mathematical society and Egyptian sydicate of scientific professions., he is a member of some research projects in Asyut university, Egypt. He is a Manager of the project Medical Diagnostic System for Endemic Diseases in Egypt Using Self Organizing Data Mining.

Sherif T. Amin, Education: M.Sc., the University of York, York, England, Computer Science Dept. June 1995. B.Sc. The University of Asyut, Asyut, Egypt, Mathematics Dept. June 1989. Professional Experience: Assistant lecturer at the dept. of Mathematics, faculty of Science, Asyut University 1995-present. Taught graduate & undergraduate courses in Mathematics Computer (Architecture, Security, and Programming). Worked for many years as System Architect and Developer for large scale technological projects in which development of a machine vision technique to be used to find a suitable solution to map matching problems in order to be incorporated into a Neural Network Memory was required. Worked as a Chief Technology Officer and assistant project manager for a national winner Thin Client e-learning system.