# A Biometric Key Generation Method Based on Semi supervised Data Clustering

[1]SAMARENDRA SAMAL,
*Gandhi Institute of Excellent Technocrats, Bhubaneswar, India*
[2]SIMANU SMARAK BEHERA,
*Gopal Krishna College of Engineering and Technology, Koraput, Odisha, India*

***Abstract*—Storing biometric templates and/or encryption keys, as adopted in traditional biometrics-based authentication methods, has raised a matter of serious concern. To address such a concern, biometric key generation, which derives encryption keys directly from statistical features of biometric data, has emerged to be a promising approach. Existing methods of this approach, however, are generally unable to appropriately model user variations, making them difficult to produce consistent and discriminative keys of high entropy for authentication purposes. This paper develops a semisupervised clustering scheme, which is optimized through a niching memetic algorithm, to effectively and simultaneously model both intra- and interuser variations. The developed scheme is employed to model the user variations on both single features and feature subsets with the purpose of recovering a large number of consistent and discriminative feature elements for key generation. Moreover, the scheme is designed to output a large number of clusters, thus further assisting in producing long while consistent and discriminative keys. Based on this scheme, a biometric key generation method is finally proposed. The performance of the proposed method has been evaluated on the biometric modality of handwritten signatures and compared with existing methods. The results show that our method can deliver consistent and discriminative keys of high entropy, outperforming-related methods.**

***Index Terms*—Biometric authentication, feature evaluation, handwritten signature, memetic algorithm, semisupervised clustering.**

## I. INTRODUCTION

**RELIABLE and secure authentication methods are critical for secure systems. Classical authentication schemes based on the token (e.g., key) or secret information (e.g., password) [45] are unable to meet stringent security demands, as they cannot differentiate between authorized users and persons who fraudulently obtain the token or secret information. Recently, biometrics-based authentication [28], [31], [43], [52], which can overcome this limitation, has received tremendous attention for verifying the identity of a person. In addition, biometrics enjoy advantages for being natural, convenient, and more importantly improbable to be lost or forgotten, making it a potential**
**replacement of the token or secret knowledge for identity verification.**

**Despite its obvious advantages, the traditional biometricsbased authentication approach can pose various security and privacy issues [29], [55]. In this approach, a biometric template, i.e., physiological and/or behavioral characteristics, of each user is required to be stored along with the encryption key, user name, and access privileges, etc., during the enrollment process. Then, during the verification process, the stored template is matched against the query biometric data provided by the user and the key can be released upon a successful matching. By storing templates, the systems are usually vulnerable to potential security breaches from malicious and spoof attacks. Once stored templates are compromised, attackers can fabricate physical spoof samples or replay compromised templates to the matcher module to gain unauthorized access. Details of such efforts can be found in [1], [19], and [58]. Moreover, stored biometric templates also raise privacy issues, as they can disclose sensitive information about the users' personality and health status [47], which can be used to profile them. Additionally, the uniqueness characterizing the biometric data and the fact that biometrics are permanently associated with the users could be exploited to track their activities enrolled in different biometric systems.**

**To address the above issues, many alternative biometricsbased authentication methods have been developed in the literature, prominent among which is the biometric key generation approach [9], [12]–[14], [17], [25], [32]–[35], [37], [40], [46], [61], [62], [65]–[69], [74]. Given a set of statistical features extracted from biometric samples of a user, existing methods of this approach typically construct the feature space by quantizing it into a number of intervals. Subsequently, each feature of the user is mapped to a short binary string individually according to the label of the interval of which the feature value is enclosed. Eventually, every individual binary string is concatenated to form the encryption key for authentication purposes. The advantage of such an approach is that the encryption keys can be generated dynamically, and neither templates nor keys are required to be stored. Generally, for this approach to be successful, the main challenges lies in effectively modeling both intra- and interuser variations of the features, thereby generating consistent keys for the same user and different keys for different users. Moreover, from the security**

perspective, the modeling scheme should be able to support the generation of such keys with high entropy. However, existing methods of this approach either largely ignore interuser variations or have a limited capability to model both intra- and interuser variations. As a result, they are generally unable to produce consistent and discriminative keys. In this paper, we propose a novel method to generate encryption keys directly from statistical features of biometrics. In our method, a semisupervised clustering scheme, which is optimized via a niching memetic algorithm (NMA) to derive optimal or near-optimal clustering solutions, has been developed to effectively and simultaneously model intra- and interuser variations. The developed semisupervised clustering scheme is employed to model the user variations on both single features and feature subsets in order to recover a large number of consistent and discriminative feature elements for key generation. Moreover, the semisupervised clustering scheme is designed such that it outputs a large number of clusters, therefore further assisting in long key generation. Based on the modeling results, we then select a set of consistent and discriminative feature elements without overlapping to generate the key for each user. The effectiveness of the proposed method has been investigated with the application to biometric of handwritten signature, which is a physically and universally accepted biometric for authentication, and compared with related work. The results show that our method can deliver consistent and discriminative keys of high entropy and outperform related methods.

The rest of this paper proceeds as follows. In Section II, we present the details of the proposed method. Then, a review and discussion of previous related work are provided in Section III. This is followed by experimental studies in Section IV. Section V concludes this paper.

## II. Proposed Method

In this section, we present our method to generate encryption keys directly from statistical features of biometric data. Fig. 1 outlines the process of the proposed method. In this method, a semisupervised clustering scheme is introduced to effectively and simultaneously model intra- and interuser variations. In the following, we first provide the details of the semisupervised clustering scheme in Section II-A. Then, a NMA for the semisupervised clustering is provided to identify optimal or near-optimal solutions in Section II-B. Finally, how the NMA-based semisupervised clustering is employed to quantize the feature space and how the keys are generated are described in Sections II-C and II-D, respectively.

### *A. Semisupervised Clustering Scheme*

Consider a set of $n$ vector $X = \{x_1, x_2, \ldots, x_n\}$ to be clustered, where $x_i$ is a vector of $d$ real-valued measurements describing statistical features extracted from each biometric sample of each user. The hard clustering [10], as we considered here, seeks for a set of clusters $C = \{C_1, C_2, \ldots, C_k\}$ with the properties: 1)and 3) $C_i C_j = \emptyset$, $iC=i=j$, $1\emptyset$, $1 \leq \leq i \leq \leq k$; 2) $ki=1\ Ci = X$; $i, j\ k$. Additionally, the clusters should reflect the structure of data such that objects in the same cluster are similar to each other while objects from distinct clusters are different from each other. This is typically achieved by optimizing a specified unsupervised criterion [2], [5], which is devised without incorporating any label information of the data. By optimizing these criteria, data objects from the same user, however, are highly possible to be grouped into different clusters. The clustering results thus could be ineffective for modeling intrauser variations. This renders the unsupervised criteria ineffective for the purpose of deriving consistent and long keys for each user. On the other hand, the data objects can be modeled based solely on their label information. In this case, however, the interuser variations are unable to be modeled, leading to the difficulty of generating discriminative, and long keys for different users. Further, by employing solely the label information for modeling, the results could be potential exposure of the user's genuine measurements.

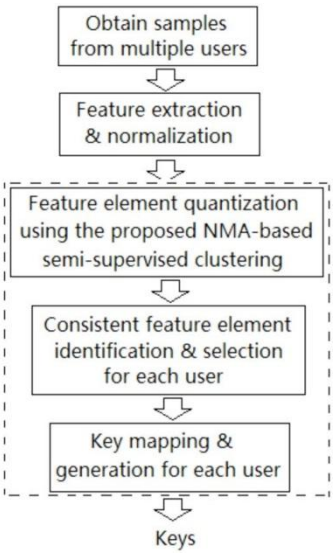



Fig. 1. Process of the proposed key generation method. The key steps are shown in dashed box.

To effectively and appropriately model both intra- and interuser variations, here, we design a semisupervised function by incorporating the label information into a clustering criterion. The designed function is based on the within-cluster and between-cluster scatter matrices, which have been popularly used in unsupervised clustering criteria [10]. One criterion is the trace($S_w^{-1}S_b$), in which the within-cluster variation $S_w$ is calculated as

$$S_w = \frac{1}{n}\sum_{j=1}^{k}\sum_{i=1}^{n} z_{ji}(x_i - m_j)(x_i - m_j)^T. \quad (1)$$

Here, $S_w$ measures the compactness of clusters, i.e., how scattered the data objects are from their cluster means, where $z_{ji} = 1$ if $x_i$ belongs to cluster $j$, 0 otherwise and $m_j$ is the mean of cluster $j$. The between-cluster variation $S_b$ is computed as

$$S_b = \sum_{j=1}^{k} \frac{n_j}{n}(m_j - m)(m_j - m)^T \quad (2)$$

and measures how scattered the cluster means are from the sample mean, where $n_j$ is the total number of objects in the cluster $j$ and $m$ is the sample mean. In the criterion of trace($S_w^{-1}S_b$), the between-cluster variation $S_b$ is normalized by the within-cluster variation $S_w$. As a result, large values of the criterion correspond to compact and well separated clusters. This criterion is invariant under any nonsingular linear transformation and can be served as an adequate basis for designing our function.

The trace($S_w^{-1}S_b$) criterion works in an unsupervised fashion. By optimizing this criterion, data objects from the same user are highly possible to be grouped into different clusters. It has been shown that, by incorporating supervised information, the clustering process can produce better models and more accurate clustering results [26], [75]. The technique of exploring supervised information to improve the clustering process is usually termed as semisupervised clustering [72]. In the semisupervised clustering, the supervised information can be expressed as pairwise constraints indicating whether a certain pair of data objects should belong to the same (must-link) or different (cannot-link) clusters. This knowledge is then being incorporated to modify clustering criteria so that they include the satisfaction of constraints. Several of such criteria have been proposed in the semisupervised clustering [22], [76]. However, they are generally developed with the purpose to seek one cluster per category of the data. Rather than seeking one cluster per category, our goal here is to identify the clustering solution, which can appropriately model intra- and interuser variations. In this sense, we formulate a function by taking into account both the trace($S_w^{-1}S_b$) criterion and must-link pairwise constraints to search for clusters such that data objects from the same user are grouped in the same cluster while the separation among clusters are well preserved.

Specifically, let $M$ be a set of must-link pairs where $(x_i, x_j) \in M$ implies $x_i$ and $x_j$ should be in the same cluster and $\omega_{ij}$ be the penalty cost for violating the constraint in $M$. We can write the function as

$$F_1 = \text{Norm}\left(\text{trace}\left(S_w^{-1}S_b\right)\right) - \text{Norm}\left(\sum_{(x_i,x_j)\in M} \omega_{ij} l\right) \quad (3)$$

where $l$ is the indicator function, $l = 1$ if objects $x_i$ and $x_j$ are assigned to different clusters, otherwise $l = 0$. If we assume uniform constraint costs $\omega_{ij}$, all constraint violations are treated equally. Intuitively, the penalty for violating a mustlink constraint between nearby points should be higher than that of between distant points. Further, the penalty should be higher if two separated must-link points are far apart from their corresponding cluster center. To reflect this intuition, we define $\omega_{ij}$ as

$$\omega_{ij} = \|x_i - x_j\|^2 \times \left(\|x_i - m_i\|^2 + \|x_j - m_j\|^2\right) \quad (4)$$

where $m_i$ and $m_j$ denote the centers of clusters for which the data objects $x_i$ and $x_j$ are assigned. Note that the values of the terms of trace($S_w^{-1}S_b$) and $\sum_{(x_i,x_j)\in M} \omega_{ij} l$ in the function $F_1$ have different order of magnitude. By optimizing this function, the result will be dominated by the term with large values. In order to closely reflect the equal importance of these two terms, they should be expressed in units of approximately the same numerical values. In this paper, since the function will be optimized via aNMA, these two terms will be dynamically normalized during evolution of the algorithm. The details of this will be described in Section II-B.

In the function $F_1$, the first term Norm(trace($S_w^{-1}S_b$)) is biased toward increasing number of clusters. The second term, on the other hand, tends to decrease the number of clusters. These two terms will thus compete with each other to form clustering solutions. However, it can be found that the solutions delivered by optimizing this function usually have a small number of clusters. To assist in generating long keys, however, we generally prefer solutions with a large number of clusters. For this purpose, we add a penalty term of $(k-1)/(k_{\max}-k)$ to $F_1$. The resulting function, denoted as $F_2$, becomes

$$F_2 = \left(\text{Norm}(\text{trace}(S_w^{-1}S_b)) - \left(\sum_{(x_i,x_j)\in M} \omega_{ijl}\right)\right) * \left((k-1)/(k_{\max}-k)\right) \quad (5)$$

Norm

where $k$ is the number of clusters encoded the solution. This penalty term is set to discourage the solutions with small number of clusters. By maximizing the function of $F_2$, we therefore aim to attain solutions with large number of clusters such that data objects from the same user are grouped in the same cluster while the separations among clusters are well preserved. It should be noted that this penalty term is determined empirically and there may be another more effective penalty term, which could result in even better performance.

### *B. NMA-Based Optimization*

As the designed function is discontinuous and nonconvex, optimizing the function to identify the optimal or near optimal partitioning of a data set with nontrivial size is a difficult problem. We further present here a NMA to approach a solution to this problem. The general procedure of the algorithm is shown in Algorithm 1.

The NMA can be viewed as an extension of the traditional GA (TGA) [11], [20]. Unlike TGA, some kind of niching selection and niching replacement operations are generally used in the NMA for selecting parent pairs and generating new population, respectively. These niching operations are employed to preserve the population diversity during the search and permit the algorithm to investigate many peaks in parallel. As a result, premature convergence, which is an intrinsic drawback when applying TGA to deal with complex optimization problems, can be usually prevented. Further, in the NMA, a certain type of local search operation is typically incorporated to improve the search efficiency. Compared with TGA, NMA can therefore be used to optimize the designed function for semisupervised clustering.

To apply NMA for semisupervised clustering, we first need to generate a population of initial solutions. This population is then undergoing an NMA-based evolution, guided by the designed function for fitness computation. The output of the algorithm is the best clustering solution in the terminal population. In the following, we first describe how the initial solutions are created. This is followed by explaining how Algorithm 1 General Procedure of the NMA-Based Semisupervised Clustering Algorithm

1) Randomly initialize a population of $P$ solutions with different numbers of clusters using a real-value-based representation.
2) Calculate the fitness value for each solution in the initial population according to (5).
3) Repeat the following sub-steps (1)–(4) until the stopping criterion is satisfied.
   a) Select parent pairs utilizing a niching selectionoperation. Repeat the procedure until $P/2$ pairs are selected.
   b) Generate offspring by employing a modified one-point crossover followed by the Gaussian mutation operation.
   c) Compute the fitness for each offspring accordingto (5).
   d) Applying a niching replacement operator to select an opponent for each offspring from the population. If the offspring has better fitness, then replace its opponent in the population.
4) Output the best solution in the terminal population.

---

the fitness of solutions is calculated using the designed function and how the population evolves to identify the optimal or near-optimal clustering solution.

*1) Population Initialization:* The proposed semisupervised data clustering involves determination of the proper number of clusters as well as appropriate clustering of the data sets. Further, for optimization problems in continuous domain, the real-value-based representation is generally preferred to encode solutions in evolutionary algorithms, as it offers higher precision than other alternative representations [44]. We therefore employ a real-value-based variable length string representation to encode solutions with different number of clusters. Specifically, each solution $i$in the initial population is encoded with a vector of $k_i \times d$real numbers, where $k_i$ denotes the number of clusters and $d$ is the dimension of data. The first $d$ values represent the center of first cluster, the next $d$ values represent that of second cluster, and so forth. The initial values of each solution are constructed via randomly assigning real numbers to each of the $d$ attributes of $k_i$ cluster centers. The values are restricted to be in the range of attribute for which they are assigned. The initial number of cluster$k_i$ is set randomly within the range of 2 to $k_{\max}$, and $k_{\max}$is the upper bound of cluster number in a data set. Here, the $k_{\max}$is calculated to be $\sqrt{n}$, which is a rule of thumb used in the clustering [51]. The number of clusters in the initial population will thus range from 2 to $k_{\max}$.

*2) Fitness Function:* The fitness of a solution indicates the goodness of result it represents. Here, the designed function $F_2$ will be employed for the fitness calculation. This function, however, consists of two terms (i.e., trace$(S_w^{-1}S_b)$ and

$\sum_{(x_i,x_j)\in M} \omega_{ijl}$ ), which should be normalized to have equally importance. Unfortunately, there is no *a priori* information available for such a purpose. In this paper, we propose to normalize their values dynamically during the NMA evolution, as follows. For each individual solution, a fitness vector, which is associated with the individual, is introduced to store the values computed from each of the two terms. Before calculating the fitness of the individual, values of the corresponding terms stored in its associated fitness vector are firstly extracted and each of them is subsequently normalized according to

$$\text{Norm}(f(x)) = \left(f(x) - f^{\min}(x)\right) \Big/ \left(f^{\max}(x) - f^{\min}(x)\right) \quad (6)$$

where $f^{\min}(x)$ and $f^{\max}(x)$ are minimum and maximum values, respectively, of the term recorded so far during evolution. Finally, the fitness of an individual can be calculated using the normalized values. The calculated fitness value will not be permanently associated with the individual solutions rather, it will be dynamically computed when needed.

*3) NMA-Based Evolution:* Based on the above initial population, an NMA-based evolution process will then attempt to optimize the fitness function $F_2$. At each generation, parent pairs are first selected from the existing population by employing a variant of multiniching crowding (VMNC) technique [21]. In the VMNC, during selection, one individual, $P_1$, is selected randomly from the population and its mate

is selected from a group of individuals of size $S$, picked randomly from the population. The individual in $S$, which is most similar to $P_1$, is chosen as its mate $P_2$. These selected pairs are subsequently crossed and mutated to generate offspring. The crossover operation is set to exchange information between each pair of parents. During crossover, we consider cluster centers encoded in the solution to be indivisible, i.e., crossover points are restricted to lie in between two cluster centers. For this purpose, we employ a crossover operation, which is analogous to the traditional one-point crossover [20], as follows. Suppose parent individuals $P_1$ and $P_2$ encode $k_1$ and $k_2$ cluster centers ($k_1 \le k_2$), respectively. The position of crossover point $x_1$ in $P_1$ is first randomly generated between 1 and $k_1$. The position of crossover point $x_2$ in $P_2$ is then randomly generated between $x_1$ and $k_2$. After that, the data segments between $x_1$ and $x_2$ in $P_1$ and $P_2$ are exchanged. The crossover will be applied stochastically on each parent pair. After crossover, the Gaussian mutation [3], which adds a unit of random Gaussian distributed value to the selected attribute, is applied with a low probability to the offspring. The value of new attribute is clipped in case it falls outside the upper or lower bound of that attribute.

To improve the time efficiency, the $k$-means algorithm [39] is further employed to refine the solution encoded in the offspring. Given an initial set of $k$ cluster centers, the $k$-means

algorithm attempts to minimize the sum of squared error (SSE) clustering criterion by iterating between two steps: assignment and update. During assignment step, each data object is assigned to a cluster whose center is closest to it. During update step, the center of each cluster is updated by the mean of data points assigned to it. The process of the algorithm converges when no further reassignment improves the SSE and is known to be very efficient. However, the performance of $k$-means is sensitive to initial cluster centers. Rather than applying $k$-means on the offspring until convergence, here, a single iteration of it will be used to refine cluster centers

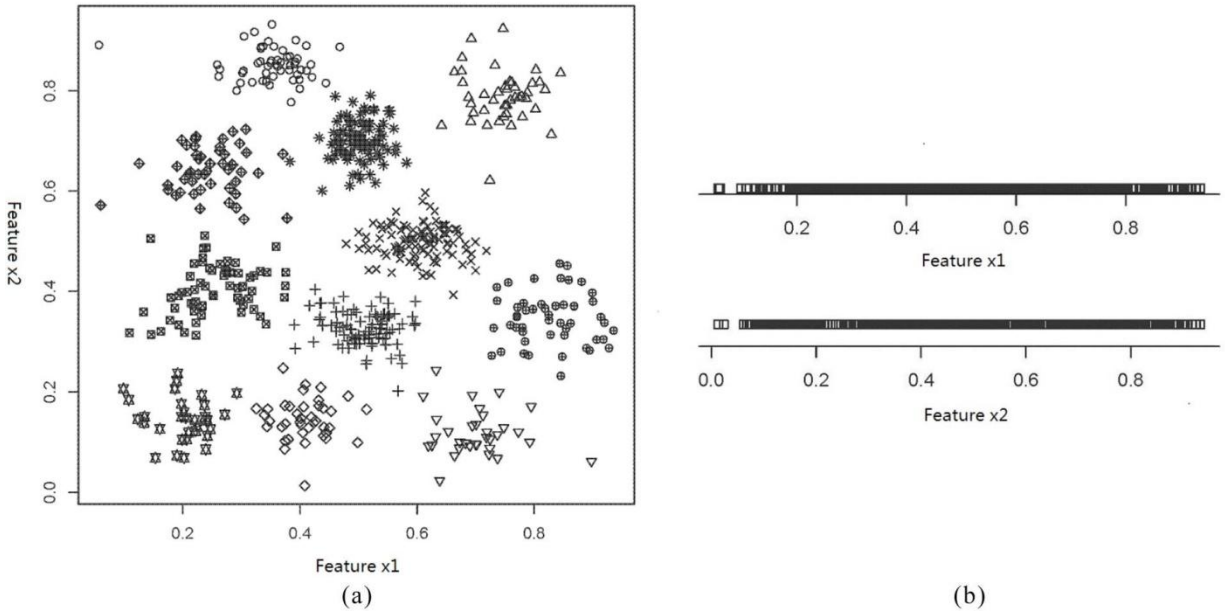



**Fig. 2.Advantage of applying the clustering on (a) feature subset over (b) single feature, whereby a clear separation among the clusters can be created v features *x*1 and *x*2 are processed together.**

**encoded in the offspring. This is done by assigning each data object to the nearest cluster center encoded in the offspring. After that, each cluster center is updated as the mean of data objects assigned to it.**

**After the *k*-means algorithm-based local operation, the niching replacement procedure of the VMNCis finally performed on the offspring. In this replacement, the offspring is paired with the most similar individual from a group of *R* solutions, which are randomly selected from the population. The offspring subsequently compete with its paired opponent to survive based on the fitness. If the offspring has better fitness, its paired opponent in the population is replaced by the offspring. The above evolution process will be repeated to optimize the fitness until the fitness of the best solution in the population has not improved for *g* generations.**

***4) Parameter Settings:*** **The above NMA-based semisupervised clustering algorithm has several parameters, which need to be set. These include crossover and mutation probabilities, the values of *S* and *R* used in the VMNC, the population size and the number of *g* for terminating the evolution process. We set the crossover and mutation probabilities to be 0.9 and 0.01, respectively. Good values for these parameters are from [0.6, 1.0] and [0.005, 0.02], respectively. The values of *S* and *R* are set to be 5 and 10, respectively. Population size equals 50 and the number of *g* in the stopping criterion is set to be 20. A larger value of either *S*, *R*, *g*or population size may lead to a longer runtime but with no significant improvement in performance. These parameters are determined experimentally on above data sets based on the average fitness value resulted from five trials of the algorithm. No attempt has been made to optimize parameter values in an absolute sense. It is thus certainly possible that a more effective set of parameter values could be found which would result in even better performance. However, it should also be noted that employing inappropriate parameter values, e.g., a too small value of mutation probability and/or population size, would degrade the overall performance of the algorithm.**

### *C. Feature Quantization*

**Feature quantization starts by taking a few biometric samples from each candidate user and calculating their feature values. Based on the obtained feature values, i.e., training data, we apply the proposed NMA-based semisupervised clustering algorithm to partition it into clusters. As data objects coming from the same user are typically similar with each other, they tend to be assigned to the same cluster. This is particularly true when features values are consistent and their data are clustered using the proposed algorithm, which considers the label information of the data. Conversely, data objects grouped into different clusters are often coming from different users. Thus, the clustering results can be utilized to model both intraandinteruser variations.**

**Generally, we can quantize the feature space by partitioning the data on each single feature. However, it is usually difficult to identify enough consistent single features to generate long keys. In addition, by quantizing each feature in isolation of other features, the algorithm may destroy important interactions among the features and cause them to perform suboptimally. To cope with such situations, we take into account the interdependence among features in the quantization process by applying the**

proposed algorithm also on feature subsets of the training data. By considering feature subsets, clustering solutions can usually offer great flexibility in the shape of formed clusters, which could better adapt to the object distribution within each cluster. This can be particularly useful to recover consistent feature subsets from inconsistent single features. Fig. 2 depicts an exemplary scenario that favors the feature subset quantization over the single feature owing to the ability in preserving the separation among clusters in a clear-cut manner. Note, we consider feature subsets with the dimension up to three in this paper. Nevertheless, it is possible that more consistent feature subsets could be found by considering feature subsets of larger dimensions.

As a result of the quantization process, we can obtain a list of single features and feature subsets (which are referred to as feature elements in this paper) as well as their corresponding clustering results, represented by the centers of clusters. Having obtained such results, we can then quantitatively distinguish among the feature elements according to their association degrees to the corresponding clusters. This is done by, for each feature element of each user, calculating its association degree, *ad*, as $ad = d_{avg}/d_{avg}$. Here, $d_{avg}$ and $d_{avg}$denote the average distance of the user's data objects and all the data objects, respectively, in the corresponding cluster to its center. Such a calculation is carried out only when all the user's data objects are assigned to the same cluster on corresponding feature element. Otherwise, its association degree is simply set to zero. The computed association degree is finally utilized to measure the consistency of feature element for that particular user. Larger values imply higher consistency.

### *D. Feature Selection and Key Generation*

Having acquired the consistency information of feature elements, the most consistent feature elements can subsequently be selected for each user to generate the key. During the selection, a trade-off exists between the consistency and length of keys, which can be controlled by varying the selection threshold. A higher threshold will result in more consistent keys. However, the length of generated keys will be decreased correspondingly. Conversely, if the threshold is decreased to produce long keys, then the consistency of keys will decrease accordingly. In principle, the selected feature elements should contain no overlapping features, which mean no features should be used more than once to generate the key for each user. Otherwise, such information could be exploited to guess the key. In order to produce long keys under this principle, the feature elements with low dimensions will be always considered first during the feature selection.

Once the feature elements are selected for each user, we can finally generate the key. For each feature element of training data, a number of clusters can be generally identified and each cluster is indexed with an incremental binary value (i.e., key-bits). For example, suppose the *i*th selected feature element and assume there are ten clusters identified on this feature element. Then, these ten clusters are indexed from 0000 to 1010. For each selected feature, when a live sample of the user is available, its value is extracted and evaluated against the centers of clusters resulted on the corresponding feature element to determine its membership and key-bits. Every selected feature element is considered in this way and the obtained individual key-bits are eventually concatenated to form the key for authentication purposes.

## III. Previous Related Work

The idea of direct key generation from biometric data was probably first raised in [6]. Following this idea, a number of methods have been proposed. These methods try to generate keys from statistical features of biometric data and can be divided into two broad categories: userdependent [9], [13], [25], [35], [40], [62], [69] and user-independent [12], [14], [17], [32]–[34], [37], [46], [61], [65]–[68], [74] methods. The user-dependent methods (UDMs) utilize the user-specific feature distribution to generate the keys. For example, Hao and Chan [25] proposed such a method. In this method, a genuine interval $[\mu-r\sigma,\mu+r\sigma]$ is first determined according to the mean, $\mu$, and standard deviation, $\sigma$, of the feature distribution for each feature of a user, together with a free parameter *r*. The remaining intervals with the same width of $2r\sigma$ are constructed from the genuine interval outwards. Then, each interval is labeled with a binary string. At the time of authentication, extracted feature values from the live biometric sample are compared with the threshold values determined previously to obtain binary strings, which are then concatenated to form the key. Similar methods are followed by Chang*et al.* [9], Vielhauer [69], Sheng *et al.* [62], and Makrushin*et al.* [40]. Chen *et al.* [13] proposed a method in which the genuine interval is constructed to accommodate the user-specific likelihood-ratio in each feature and the remaining intervals are created in an equal probable manner. Kumar and Zhang [35] employed an entropy-based scheme to reduce class impurity/entropy in the intervals by recursively splitting every interval until a stopping criterion is met. The final intervals will be resulted in such a way that the majority samples enclosed within each interval would belong to a specific user. By taking care of intrauser variations of biometric features, UDMs can offer good performance in generating consistent keys for the same user. However, they largely ignore interuser variations, thus lacking the capability to derive discriminative keys for different users. Another critical problem of UDMs is the potential exposure of user's

genuine measurements, since constructed intervals serve as a clue by which the user's measurements could be located.

The user-independent methods instead utilize the background feature distribution to generate keys. A number of such methods have been proposed in the literature. Monrose*et al.* [46], Teoh *et al.* [65], and Verbitskiy*et al.* [68] employed a scheme that quantizes each background feature space into two intervals (each interval is labeled with a single bit "0" or "1") based on a prefix threshold. A feature value that falls into an interval is mapped to the corresponding 1-bit output label. Tuyls*et al.* [67] and Kevenaar*et al.* [34] introduced a similar 1-bit quantization scheme, but instead of prefixing the threshold, the mean of background feature distribution is used as the threshold for each feature. As a result, both intervals contain 0.5 background probability mass. The interval that the genuine user is expected to fall into is referred to as the genuine interval. Kelkboom*et al.* [32], [33] analytically expressed the genuine and imposter bit error probability and subsequently developed a framework to estimate the genuine and imposter Hamming distance probability mass functions of a biometric system. This framework is based on a 1-bit equal-probable quantization scheme under the assumption that both intrauser and interuser variations are Gaussian distributed. The above methods all employ the technique to produce a 1-bit output per feature and are generally unable to generate long keys.

While long keys are desired, research attention has then shifted to explore techniques, which can derive multibits from the feature space. Yip *et al.* [74] presented a method in which the space of each feature is segmented into a number ofequalwidth intervals in accordance with the quantity of bits required to be extracted from the feature. Teoh *et al.* [66] put forward a method to search for a multibits assignment for each feature. Specifically, the space of each feature is initially segmented into $2n$ equal-width intervals with $n$ denoting the intended number of bits to be allocated. Twice the standard deviation of estimated probability density function (pdf) is then taken as the evaluation measure, determining whether the width of constructed interval is sufficiently large to accommodate such a pdf. With incremental $n$, the procedure is repeated iteratively until the optimal $n$ is found. Chen *et al.* [14] introduced a scheme to determine the number of segmented spaces of each feature by optimizing the detection rate. The detection rate refers to the maximum user probability mass captured by an interval over a single feature space. Similar multibits allocation scheme has also been developed in [12], but by optimizing a different bit-allocation measure: area under the false rejection rate (FRR) curve. Lim *et al.* [37] designed a method using an integrated bit reliability and feature signal to noise ratio in performing the multibits allocation. In addition, multibits allocation algorithms have also been proposed by considering clustering and vector quantization techniques. Fairhurst *et al.* [17] devised a framework, in which the traditional $k$-means clustering algorithm is applied to partition the feature space into a number of segments. This framework is later extended in [61] by proposing a more effective clustering method to create segments induced in the features. Yamazaki and Komatsu [73], on the other hand, applied the vector quantization method for segmenting the space of each feature. The user-independent approach is global, i.e., every user employs the same quantization setting, and thus does not require the user-specific data to be stored. Although this approach could prevent undesired leakage of the user-specific information, the overall authentication performance of existing methods can rarely be encouraging. This is mainly due to they usually have limited capability to model both intra- and interuser variations, and thus are difficult to produce consistent and discriminative keys.

Apart from the above approach, many alternative techniques have also emerged to address issues arising for template and/or key storage in biometrics-based authentication systems. These methods can be loosely divided into two categories: biometric key binding and feature transformation. The biometric key binding-based methods [8], [48], [54], [59], [63] involve binding an encryption key to the enrollment biometric which can be recovered with a legitimate probe biometric sample. These methods provide solutions in that both the key and biometric template are inaccessible to attackers, while the key can be released with a successful biometric matching. However, they require "fuzzy" biometric matching to be performed in the encrypted domain, a task that is very difficult to implement.

Firstly, it is difficult to develop a meaningful metric to measure the similarity in encrypted representations. Further, biometric matchers of this technique usually assume the query sample and template are well aligned. However, finding the appropriate alignment between the query sample and template in the encrypted domain is also difficult. The feature transformationbased methods [16], [36], [38], [49], [50], [56], [64], on the other hand, employ a transformation function to extract a new biometric template. Depending on the characteristics of the transformation function, the template protection mechanism either lets the invertible transformation function recover the original biometric template [16], [50], [64] or applies a oneway noninvertible hash function to conceal the biometric trait [36], [38], [49], [56]. The template can be "canceled" by using another function, if it is compromised. For the invertible transformation function-based methods, a function dependent on some

parameters, which can be used as a key, is applied to the input biometrics. The security of these methods mostly relies on the security of key distribution. The loss or disclosure of the key, therefore, results in serious security issues [57]. For the noninvertible function-based methods, it is computationally hard to retrieve the original data from produced templates. However, due to the characteristics of biometric features, it is difficult to design appropriate transformation functions. For example, in [36], noninvertible functions are applied to face images to obtain transformed templates, which, however, allow human inspection. In [56], polar, Cartesian and functional noninvertible transformations are utilized to modify the fingerprint minutiae template. However, only a small fraction of the resulting data is in practice noninvertible [7] and the scheme is vulnerable to a record multiplicity attack, where an adversary who acquires two or more distinct stored templates is able to recover the original information [53].

It should also be mentioned that related GAs or MAs [4], [27], [42] have been proposed in the literature. However, these algorithms are designed either for data clustering by optimizing a specified unsupervised criterion or semisupervised clustering, which seeks one cluster per category of the data. Further, they may have difficulty in delivering the optimal or near-optimal solution corresponding to the given data set as the mechanism to preserve population diversity is generally lacked in these algorithms.

## IV. Experimentation

In this section, we evaluate the proposed method and compare it with related methods. After describing the data sets used in experiments in Section IV-A, we describe the performance indexes in Section IV-B. This is followed by experimental results with a comparative analysis in Section IV-C.

### *A. Data Sets*

The biometric modality of handwritten signature has been used to evaluate the proposed method. The advantage of adopting handwritten signature over other biometric modalities is that it is minimally intrusive and has high level of user acceptance. The signature, however, also exhibits a major disadvantage of that it usually shows high intrauser variations [17]. As such, the signature data can be regarded to be a near-worst case for which the proposed method can be evaluated, and thus gives a useful indication of its capability.

The signature data sets used in experiments were collected in a typical practical environment from the general public during trials of an automatic signature verification system [17]. These signature samples are captured using an A4-sized graphics tablet over multiple sessions. The overall data set comprises of 7430 signatures donated by 359 volunteers. The number of signature samples collected from each volunteer at each trial is not fixed due to practical reasons. Three subsets, $DB_1$, $DB_2$, and $DB_3$ have been selected for our evaluation purpose. The $DB_1$ contains 1000 signatures from 100 volunteers while the $DB_2$ and $DB_3$ have 2000 and 2500 signatures from 200 and 250 volunteers, respectively. $DB_2$ is a superset of the $DB_1$ while $DB_3$ is superset of the $DB_2$. To ensure sufficient training data, the signatures in the subsets are from volunteers who contribute at least ten samples and the first ten samples are used in the experiments.

Many statistical features can be derived [18], [23], [24] from the obtained signature data. As dynamic features of the signature are not directly accessible, here, we extract 30 of such features comprising of: the overall duration of signature, number of strokes, pen-down time, pen-up time, average pen velocity and acceleration, time of minimum/maximum pen velocity, and acceleration occurred in the vertical/horizontal direction, sum of local minima and maxima in the vertical/horizontal direction, and amount of time moving to the up, down, left, and right. Further, features involve the number of times of the following events: pen ceases to move horizontally/vertically and pen passes the mean vertical center. Additionally, seven invariant moments, measuring the number of samples, vertical/horizontal divergence, vertical/horizontal mass and vertical/horizontal imbalance of the writing [23], are considered. As feature subsets are utilized in our method, any features that can be derived from the others are excluded. For example, given the pen-down and pen-up time, a feature denoting the ratio of pen-down and pen-up time is then redundant. All the values of each extracted feature are normalized to have the average value of zero with standard deviation of one.

### *B. Performance Indexes*

The false acceptance rate (FAR) and FRR [71] are the most commonly adopted indexes for measuring the performance of biometric authentication systems. In this paper, false acceptances are instances in which correct keys are generated even though presented signatures are not from valid users while false rejections refer to attempts, which fail to regenerate correct keys despite valid users presenting their signatures. Generally, both the FAR and FRR depend on the space of generated keys. For instance, a long key usually leads to a low FAR but a high FRR. Here, we therefore evaluate the FAR and FRR against keys with increasing spaces. All FRR and FAR values reported in our results were obtained using the same experimental procedure. The FRRis computed by selecting a set of six samples randomly from each user for training and using the remaining four samples for testing.

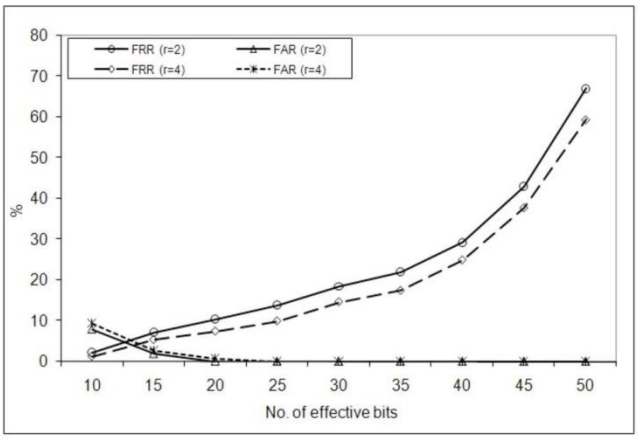
For the FAR, since no signatures from professional imitators are available, it is calculated by choosing 20 signatures randomly from other users and then tested against each user.

In the proposed method, variations of signature samples can cause the next closest cluster to become the authentic cluster. To deal with such transient errors, it is usually necessary to attempt alternative keys, which are close to each other. Here, we generate alternative keys by employing a strategy to replace a certain key-bits of the initial key with the one tagged on the next closest cluster. For instance, suppose a key = $10_1\,010_2\,101_3\,01_4\,101_5$ is the initial key derived from the five feature elements of a user's signature, one alternative key that the authentication program will attempt can then be obtained by replacing the key-bits derived on the second feature element (i.e., $010_2$) with the key-bits tagged on the next closest cluster (assuming that is $011_2$) to yield the key of $10_1\,011_2\,101_3\,01_4\,101_5$. Suppose this strategy is used to correct a number of*t* of such errors, the authentication program will therefore attempt $\sum_{i=1}^{t} w!/i!(w-i)!$ keys before returning a rejection decision, where $w$ denotes the total number of selected feature elements. The value of $t$ that can be taken is determined mainly by the time allowed to spend before issuing a negative answer. Here, $t = 2$ and $t = 4$ are evaluated in our method.

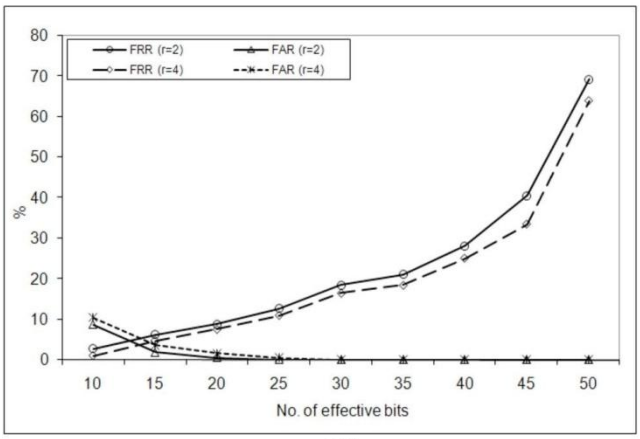
### *C. Results*

We first assess the performance of proposed method in terms of the FRR and FAR. The results are shown in Fig. 3(a)–(c) for experiments on $DB_1$, $DB_2$, and $DB_3$, respectively. Each FRR or FAR is parameterized by the number of effective bits *eb*, computed from the logarithm to base 2 of the reciprocal of probability that a person obtaining the information of clustering results of the selected feature elements can guess all authentic clusters correctly. The *eb* value of a key will be less than or on rare occasions equal to the key length as some coding spaces for the given bit length may not be used to index the clusters. For each value of *eb*, the selection threshold will be gradually reduced for each user until a key with approximately *eb* effective bits is derived. Here, we generate keys with effective bits from 10 to 50 in steps of 5 and the corresponding FRR and FAR performances are then plotted. The results on $DB_1$ show that our proposed method is capable of generating keys with good consistency, discriminatory, and security. Particularly, to generate keys of effective bits $eb = 35$ with $t = 2$, the proposed method achieves the performance of FRR= 22.0% and FAR = 0%. Experiments with $t = 4$ show an even better performance: to generate keys with the same number of effective bits, the FRR reduces to 17.5% while FAR remains 0%. Similar results can also be observed from experimental results using $DB_2$ and $DB_3$. This may provide some initial clues that its performance will not significantly degrade when the number of users is relatively large.

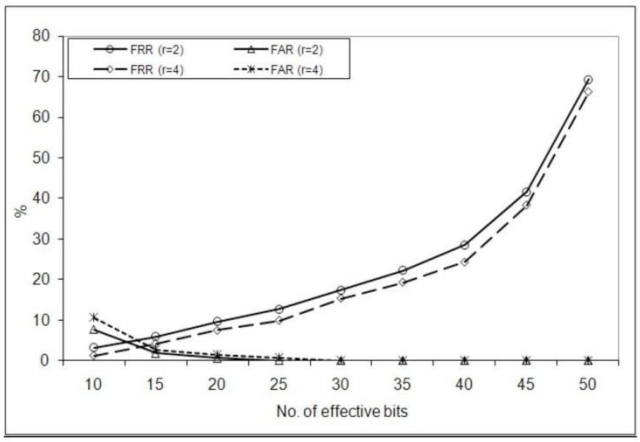
Then, experiments are carried out to evaluate the significance of semisupervised clustering function, NMA search mechanism and employment of feature subsets in the proposed method for feature quantization and key generation. For this purpose, we compared the proposed method (denoted as SSCKey) with its five variants. In the first




(a)




(b)




(c)

**Fig. 3. FRR and FAR performance of the proposed method on (a) DB1, (b) DB2, and (c) DB3 plotted against the generated keys with increasing number of effective bits.**

**and second variants (denoted as SSCKey_1 and SSCKey_2), instead of the proposed semisupervised clustering function, the Davies–Bouldin (DB) [15] clustering criterion and the proposed function $F_1$ [i.e., without the penalty term of $(k-1)/(k_{max}-k)$], respectively, are used for fitness computation. In the third and fourth variants (denoted as SSCKey_3 and SSCKey_4), rather than the NMA search mechanism, we employ the constraint *k*-means algorithm [70] and a TGA, respectively, for semisupervised clustering. In the fifth variant (denoted as SSCKey_5), only the quantization solutions of single features are considered for key generation. All other settings of the five variants are the same as the SSCKey. Before discussing comparative results, we first briefly describe the DB clustering criterion, constraint *k*-means algorithm and GA used for experiments. The DB criterion is defined as the ratio of sum of within-cluster scatter to between-cluster separation. Let $k$ be the number of clusters and $d_{ij}$ be the distance between cluster $C_i$ and $C_j$. Then, the DB criterion is defined as**

$$\mathbf{DB} = {}^{1}\!/_{k} \sum_{i=1}^{k} \max_{j,j\neq i} \{(S_i + S_j)/d_{ij}\}$$

**where $S_i$ denotes the scatter of $i$th cluster and is calculated as $S_i = {}^{1}\!/_{n_i} \sum_{x\in C_i} x - m_i$. Here, $m_i$ is the $i$th cluster center and $n_i$ denotes the number of objects within cluster $C_i$. This criterion has been popularly used for data clustering and the best solution occurs at the optimal value of the criterion. The constraint *k*-means, designed for semisupervised clustering, is a modified version of traditional *k*-means algorithm by taking into account pairwise constraints. Starting from initial cluster centers, each data object is assigned to a cluster with the closest center such that it does not violate a constraint, after that each center is updated as the center of data objects belonging to that particular cluster. This procedure is repeated until the algorithm converges. To make comparisons more meaningful, the number of clusters, $k$, which has to be fixed in the constraint *k*-means, is set to equal the number of clusters identified by our proposed method for experiments on each feature element of the training data. The GA used in the experiments employs the tournament selection, two-point crossover, and Gaussian mutation to perform semisupervised clustering. During replacement, the elitist strategy is used to generate new populations. Fig. 4(a) and (b) shows the FRR and FAR performance of the six algorithms on $DB_1$ and $DB_2$ data sets, respectively. They are run to generate keys with different number of effective bits by adjusting the feature selection threshold and without employing any error correction technique. It can be observed from Fig. 4 that the SSCKey can generally achieve the best performance across different numbers of effective bits of the generated keys. For example, to generate keys with effective bits $eb$ = 30 on $DB_1$, the SSCKey_1, SSCKey_2, SSCKey_3, SSCKey_4, and SSCKey_5 give the FRR of 29.6%, 23.6%, 24.2%, 22.8%, and 34.4%, respectively, while the SSCKey achieves the FRR at 21.4%. By examining the clustering**

results of SSCKey, SSCKey_1, and SSCKey_2, which use different criteria for fitness computation, we can find that all methods can deliver compact and well-separated clusters. However, by using the proposed semisupervised criterion, the SSCKeyis able to identify more consistent clustering solutions than the SSCKey_1 in terms of objects from the same user being grouped into the same cluster. For example, on the feature element consisting of the average pen velocity and acceleration, objects from the same user are grouped into the same cluster for over 82% of the users by applying the SSCKey on $DB_1$. By comparison, this value drops to about 71% by employing the SSCKey_1. Consequently, the semisupervised function can be used to produce clustering solutions,

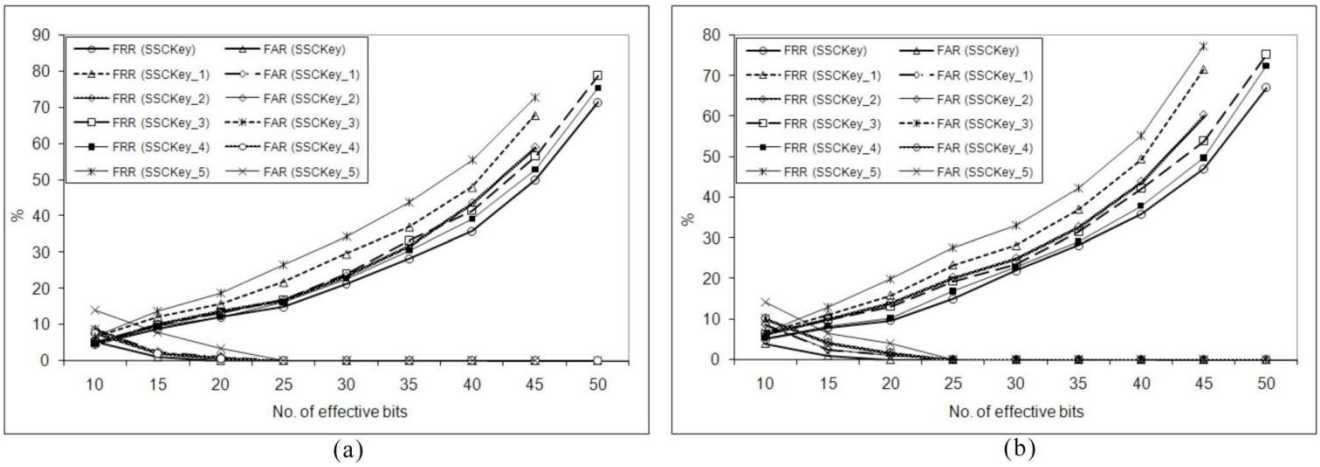



**Fig. 4. FRR and FAR performance of the proposed method and its five variants on (a) DB1 and (b) DB2 plotted against the generated keys with increasing number of effective bits.**

which can more appropriately model both intra- and interuser variations of the features. Moreover, the SSCKey is generally able to deliver clustering solutions with a large number of clusters, which can adequately support long key generation. By contrast, both the SSCKey_1 and SSCKey_2 fail to do so and thus have difficulty in producing long keys. For example, out of 30 features on $DB_1$, the keys with maximum entropy could be generated for all users having an average of about 47 and 49 effective bits by employing the SSCKey_1 and SSCKey_2, respectively, while the SSCKey can deliver such keys with around 58 effective bits on average. Looking at the results of SSCKey, SSCKey_3 and SSCKey_4, which use different semisupervised clustering techniques for feature quantization, we can see that the SSCKey_3 and SSCKey_4 generally do not perform as well as the SSCKey. This is due to the fact that both the constraint *k*-means algorithm and the GA are susceptible to sub-optimal solutions, therefore leading to relatively inferior performance in feature quantization for key generation. For the SSCKey_5, which generates keys based on quantization solutions of single features, it can be observed that its FRR performance degrades sharply with increasing effective bits of the keys. This is mainly because single features generally show high intrauser variations with few consistent ones, thus significantly reducing the performance of SSCKey_5 in producing consistent keys with high entropy. By considering the feature subsets, the SSCKey shows that many consistent feature subsets can be recovered from the inconsistent single features. This may help to understand why the SSCKey can perform better than the SSCKey_5 and to justify the importance of considering feature subsets for feature quantization and key generation.

Finally, we assess the performance of proposed method by comparing it with four previous related methods that resemble the algorithms described in [12], [61], [66], and [69], respectively. The method presented in [69] is a UDM, which generates keys by constructing a user-specific quantization setting for each user. This method segments the entire boundary of each feature into several intervals based on the user's intravariations. The segmentation is implemented as follows. First, the user's genuine interval of each feature is determined. Then, the same interval unfolds to both ends of the population generic feature boundary. The user's genuine interval is defined as $I = [I_{low} \times (1 - a), I_{high} \times (1 + a)]$, where $I_{low}$and $I_{high}$are the low and high boundary, respectively, of feature values. This interval can be adjusted by varying the parameter *a*, and a large value will reduce the FRR but increase the FAR. The second method proposed by Teoh *et al.* [66] is a user-independent method (denoted as UIM_1), which has been described previously in Section III. In this method, the generated keys are used to construct cancellable biometrics. Technically, all the keys generated via the biometric key generation approach can be used to create cancelable biometrics. Here, we address the consistency, discriminatory, and entropy of generated keys, which is

crucial to create cancellable biometrics. The third method presented in [12] is also a user-independent method (denoted as UIM_2). Given the bit-error probability, this method allocates bits dynamically to every feature in a way that the analytical area under the FRR curve for hamming distance evaluation is minimized. Since a straightforward brute force search of all possible bit assignments may incur an extremely high computational complexity, the authors employed a dynamic programming algorithm to search for the optimal bit assignment. The fourth method proposed in [61] (denoted as FCKey) employs a fuzzy clustering technique, which is optimized by a GA, for feature quantization and key generation. These five methods are evaluated using the data set $DB_1$. The UDMis implemented using a relatively large value of $a = 1$ to achieve a low FRR. Again, all the methods are implemented to deliver keys with different numbers of effective bits by varying the feature selection threshold. In principle, error correction techniques, such as the Shamir's [60] secret sharing scheme, can be conjoined with any of the five methods to improve their performance. To make a fair comparison, here, we conduct the experiments without incorporating any error correction technique in any of these methods. Fig. 5 shows the results.

Fig. 5 shows, when used to generate long keys, our proposed method is capable of producing the keys with significantly higher consistency than the four previously developed methods. For instance, to generate keys with effective bits

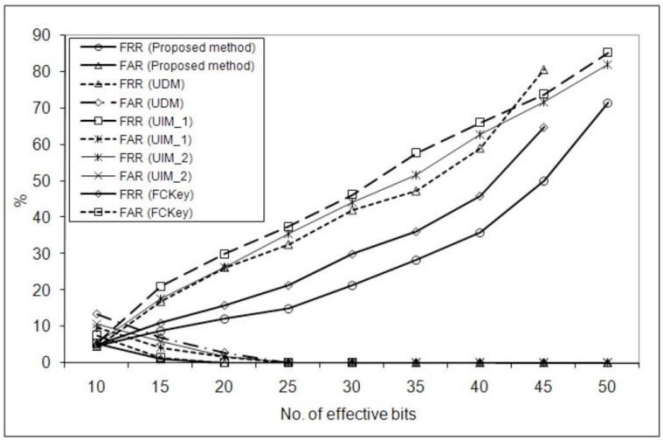



Fig. 5. FRR and FAR performance of five methods on DB1 plotted against the keys with increasing number of effective bits.

of 40, the FRR of UDM, UIM_1, UIM_2, and FCKey turns out to be 59.1%, 65.9.6%, 62.5%, and 46.3%, respectively. By contrast, our method has a much lower FRR at 36.1%. The poorer performance of the UIM_1 and UIM_2 is mainly attributed to the fact that they rely on the quantization solutions of single features, which usually show high intravariations, as well as their limited capability to model intrauser variations of the features. By taking care of the intrauser variations of the features, the UDM can generally deliver more consist keys than the UIM_1 and UIM_2. However, the quantization scheme of the UDM relies also on single features, which greatly degrades their FRR performance in deriving long keys. By modeling intra- and interuser variations using an unsupervised fuzzy clustering technique, the FCKey can achieve even better results than the UDM. However, it has limited capability to simultaneously model both intra- and interuser variations. When used to generate short keys, the results show that the proposed method can deliver the keys with comparable consistency but more discriminatively than the four methods compared. For instance, to deliver keys with ten effective bits, the UDM, UIM_1, UIM_2, and FCKey carry with the FAR of 13.5%, 9.7%, 10.8%, and 7.6%, respectively, while our method gives 5.4%. The performance of the UDM is not surprising since its quantization scheme does not consider interuser variations of the features. For the UIM_1, UIM_2, and FCKey, though they perform better than the UDM, their performances are still worse than our proposed method. Clearly, based on above experiments, our proposed method is the best alternative and can be used to generate consistent and discriminate keys of high entropy.

## V. Conclusion

This paper presents a semisupervised clustering-based method to directly extract encryption keys from statistical features of biometric data for authentication purposes. In our method, we develop a semisupervised clustering scheme, which is optimized through a NMA to effectively and simultaneously model intra- and interuser variations. The developed scheme is employed to model the user variations on both single features and feature subsets to recover a large number of consistent and discriminative feature elements for key generation. Furthermore, to assist in generating long keys, the semisupervised clustering is designed to output a large number of clusters. Such a method eliminates the need of storage of biometric templates as well as encryption keys, and can offer secure biometric authentication. The developed method has been evaluated on the biometric modality of signature data, which can be considered as the representative of a near-worst case scenario for which the method can be tested. The results indicate that it can be used to generate the keys with good consistency, discriminatory, and entropy, outperforming-related methods.

There are several directions in which this paper can be extended further. Firstly, the generated keys can be used together with the technique such as password "salting" [41] by padding with other information (e.g., pass phase,

user name, etc.) to make them even harder to decode. Further, as a general method, the proposed method is applicable to other biometrics, including voice, fingerprint, face, etc. Therefore, statistical features derived from multiple biometric modalities could be utilized to generate more secure keys. Finally, while we have demonstrated the effectiveness of our proposed method for key generation, the method is sufficiently flexible to be applied for biometric discretization. In this regard, we believe the proposed method can bring significant benefits to the biometric systems with fast matching requirements or constrained storage capability by converting biometric features into a binary string, which we plan to investigate in future.

## REFERENCES

[1] J. Adams, G. Dozier, and K. Bryant, "Neurogenetic reconstruction of biometric templates," in *Proc. IEEE Southeastcon*, Orlando, FL, USA, 2012, pp. 1–8.

[2] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. Pérez, and I. Perona, "An extensive comparative study of cluster validity indices," *Pattern Recognit.*, vol. 46, no. 1, pp. 243–256, 2013.

[3] T. Back, *Evolutionary Algorithms in Theory and Practice*. New York, NY, USA: Oxford Univ. Press, 1996.

[4] S. Bandyopadhyay and U. Maulik, "Genetic clustering for automatic evolution of clusters and application to image classification," *Pattern Recognit.*, vol. 35, no. 2, pp. 1197–1208, 2002.

[5] J. C. Bezdek and N. R. Pal, "Some new indexes of cluster validity," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 28, no. 3, pp. 301–315, Jun. 1998.

[6] A. Bodo, "Method for producing a digital signature with aid of a biometric feature," German Patent DE 424 390 8A1, 1994.

[7] T. E. Boult, W. J. Schreirer, and R. Woodworth, "Revocable fingerprint biotokens: Accuracy and security analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Minneapolis, MN, USA, 2007, pp. 1–8.

[8] F. Bui, K. Martin, H. Lu, K. Plataniotis, and D. Hatzinakos, "Fuzzy key binding strategies based on quantization index modulation (QIM) for biometric encryption applications," *IEEE Trans. Inf. Forensics Secur.*, vol. 5, no. 1, pp. 118–132, Mar. 2010.

[9] Y. Chang, W. Zhang, and T. Chen, "Biometric-based cryptographic key generation," in *Proc. IEEE Conf. Multimedia Expo (ICME)*, vol. 3. Taipei, Taiwan, 2004, pp. 2203–2206.

[10] C. A. Charu and K. R. Chandan, Eds., *Data Clustering: Algorithms and Applications*. Boca Raton, FL, USA: CRC Press, 2013.

[11] X. S. Chen, Y. S. Ong, M. H. Lim, and T. C. Tan, "A multi-facet survey on memetic computation," *IEEE Trans. Evol. Comput.*, vol. 15, no. 5, pp. 591–607, Oct. 2011.

[12] C. Chen and R. Veldhuis, "Extracting biometric binary strings with minimal area under the FRR curve for the hamming distance classifier," *Signal Process.*, vol. 91, no. 4, pp. 906–918, 2011.

[13] C. Chen, R. Veldhuis, T. Kevenaar, and A. Akkermans, "Multi-bits biometric string generation based on the likelihood ratio," in *Proc. 1st IEEE Int. Conf. Biometrics Theory Appl. Syst. (BTAS)*, Crystal City, VA, USA, 2007, pp. 1–6.

[14] C. Chen, R. Veldhuis, T. Kevenaar, and A. Akkermans, "Biometric quantization through detection rate optimized bit allocation," *EURASIP J. Adv. Signal Process.*, vol. 2009, Art. ID 784834.

[15] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979.

[16] N. Dey, B. Nandi, and M. Dey, "BioHash code generation from electrocardiogram features," in *Proc. IEEE Int. Adv. Comput. Conf.*, Ghaziabad, India, 2013, pp. 732–735.

[17] M. Fairhurst, S. Hoque, W. G. J. Howells, and F. Deravi, "Evaluating biometric encryption key generation," in *Proc. 3rd Cost 275 Workshop Biometrics Internet*, Hertfordshire, U.K., 2005, pp. 93–96.

[18] J. Fierrez-Aguilar *et al.*, "An on-line signature verification system based on fusion of local and global information," in *Audio- and Video-Based Biometric Person Authentication*. Lecture Notes in Computer Science, vol. 3546. Berlin, Germany: Springer, 2005, pp. 523–532.

[19] J. Galbally, A. Ross, and M. Gomez-Barrero, "Iris image reconstruction from binary templates: An efficient probabilistic approach based on genetic algorithms," *Comput. Vis. Image Understanding*, vol. 117, no. 10, pp. 1512–1525, 2013.

[20] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA, USA: Addison-Wesley, 1989.

[21] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in genetic algorithms," in *Foundations of Genetic Algorithms*, G. J. E. Rawlins, Ed. San Mateo, CA, USA: Morgan Kaufmann, 1991, pp. 69–93.

[22] N. Grira, M. Crucianu, and N. Boujemaa, "Unsupervised and semisupervised clustering: A brief survey," in *Proc. 7th ACM SIGMM Int. Workshop on Multimedia Information Retrieval*, pp. 9–16, 2005.

[23] R. M. Guest, "The repeatability of signatures," in *Proc. 9th Int. Workshop Frontiers Handwriting Recognit.*, Tokyo, Japan, 2004, pp. 492–497.

[24] S. Hangai, S. Yamanaka, and T. Hamamoto, "On-line signature verification based on altitude and direction of pen movement," in *Proc. IEEE Int. Conf. Multimedia Expo*, vol. 1. New York, NY, USA, 2000, pp. 489–492.

[25] F. Hao and C. W. Chan, "Private key generation from on-line handwritten signatures," *Inf. Manage. Comput. Secur.*, vol. 10, no. 4, pp. 159–164, 2002.

[26] S. C. H. Hoi, W. Liu, and S. F. Chang, "Semi-supervised distance metric learning for collaborative image retrieval and clustering," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 6, no. 3, pp. 1–26, 2010.

[27] A. K. Jain, "Data clustering: 50 years beyond *k*-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[28] A. K. Jain, A. A. Ross, and K. Nandakumar, *Introduction to Biometrics*. Boston, MA, USA: Springer, 2011.

[29] A. K. Jain and K. Nandakumar, "Biometric authentication: System security and user privacy," *Computer*, vol. 45, no. 11, pp. 87–92, 2011.

[30] P. K. Janbandhu and M. Y. Siyal, "Novel biometric digital signatures for internet-based applications," *Inf. Manage. Comput. Secur.*, vol. 9, no. 5, pp. 205–212, 2001.

[31] W. Kang and Q. Wu, "Pose-invariant hand shape recognition based on finger geometry," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 11, pp. 1510–1521, Nov. 2014.

[32] E. J. C. Kelkboom, G. G. Molina, T. A. M. Kevenaar, R. N. J. Veldhuis, and W. Jonker, "Binary biometrics: An analytic framework to estimate the bit error probability under Gaussian assumption," in *Proc. 2nd Int. Conf. Biometrics Theory Appl. Syst.*, Arlington, VA, USA, 2008, pp. 1–6.

[33] E. J. C. Kelkboom, G. G. Molina, T. A. M. Kevenaar, R. N. J. Veldhuis, and W. Jonker, "Binary biometrics: An analytic framework to estimate the performance curves under Gaussian assumption," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 3, pp. 555–571, May 2010.

[34] T. A. M. Kevenaar, G. J. Schrijen, M. Van der Veen, A. H. M. Akkermans, and F. Zuo, "Face recognition with renewable and privacy preserving binary templates," in *Proc. IEEE Workshop Autom. Identif. Adv. Technol.*, Buffalo, NY, USA, 2005, pp. 21–26.

[35] A. Kumar and D. Zhang, "Hand geometry recognition using entropybased discretization," *IEEE Trans. Inf. Forensics Secur.*, vol. 2, no. 2, pp. 181–187, Jun. 2007.

[36] H. Lee, C. Lee, J. Y. Choi, and J. Kim, "Changeable face representations suitable for human recognition," in *Advances in Biometrics*. Lecture Notes in Computer Science, vol. 4642. Berlin, Germany: Springer, 2007, pp. 557–565.

[37] M. H. Lim, A. B. J. Teoh, and K. A. Toh, "An efficient dynamic reliability-dependent bit allocation for biometric discretization," *Pattern Recognit.*, vol. 45, no. 5, pp. 1960–1971, 2012.

[38] E. Maiorana, P. Campisi, J. Fierrez, J. Ortega-Garcia, and A. Neri, "Cancelable templates for sequence-based biometrics with application to on-line signature recognition," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 3, pp. 525–538, May 2010.

[39] J. MacQueen, "Some methods for classification and analysis of multi-variate observations," in *Proc. Berkeley Symp. Math. Statist. Probability*, vol. 1. *Theory and Statistics*. Berkeley, CA, USA, 1967, pp. 281–297.

[40] A. Makrushin, T. Scheidat, and C. Vielhauer, "Improving reliability of biometric hash generation through the selection of dynamic handwriting features," in *Transactions on Data Hiding and Multimedia Security VIII*. Lecture Notes in Computer Science, vol. 7228. Berlin, Germany: Springer, pp. 19–41, 2012.

[41] U. Manber, "A simple scheme to make passwords based on oneway functions much harder to crack," *Comput. Secur.*, vol. 15, no. 2, pp. 171–176, 1996.

[42] U. Maulik and S. Bandyopadhyay, "Genetic algorithm based clustering technique," *Pattern Recognit.*, vol. 33, no. 9, pp. 1455–1465, 2000.

[43] O. Mendels, H. Stern, and S. Berman, "User identification for home entertainment based on free-air hand motion signatures," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 11, pp. 1461–1473, Nov. 2014.

[44] Z. Michalewicz, *Genetic Algorithms + Data Structure = Evolution Programs*, 3rd ed. Berlin, Germany: Springer, 1996.

[45] B. Miller, "Vital signs of identity," *IEEE Spectrum*, vol. 31, no. 2, pp. 22–30, Feb. 1994.

[46] F. Monrose, M. K. Reiter, Q. Li, and S. Wetzel, "Cryptographic key generation from voice," in *Proc. IEEE Symp. Secur. Privacy*, Washington, DC, USA, 2001, pp. 202–213.

[47] E. Mordini, "Biometrics, human body and medicine: A controversial history," in *Ethical, Legal and Social Issues in Medical Informatics*. Rome, Italy: IGI Global, 2008, pp. 249–272.

[48] K. Nandakumar, A. K. Jain, and S. Pankanti, "Fingerprint-based fuzzy vault: Implementation and performance," *IEEE Trans. Inf. Forensics Secur.*, vol. 2, no. 4, pp. 744–757, Dec. 2007.

[49] B. S. Oha, K. A. Toha, K. Choib, A. B. J. Teoha, and J. Kima, "Extraction and fusion of partial face features for cancelable identity verification," *Pattern Recognit.*, vol. 45, no. 9, pp. 3288–3303, 2012.

[50] O. Ouda, N. Tsumura, and T. Nakaguchi, "BioEncoding: A reliable tokenless cancelable biometrics scheme for protecting iris codes," *IEICE Trans. Inf. Syst.*, vol. 93-D, pp. 1878–1888, Jan. 2010.

[51] N. R. Pal and J. C. Bezdek, "On cluster validity for the fuzzy *c*-means model," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 3, pp. 370–379, Aug. 1995.

[52] P. P. Paul, M. L. Gavrilova, and R. Alhajj, "Decision fusion for multimodal biometrics using social network analysis," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 11, pp. 1522–1533, Nov. 2014.

[53] F. Quan, S. Fei, C. Anni, and Z. Feifei, "Cracking cancelable fingerprint template of Ratha," in *Proc. Int. Symp. Comput. Sci. Comput. Technol.*, Shanghai, China, 2008, pp. 572–575.

[54] S. Rane and P. Boufounos, "Privacy-preserving nearest neighbor methods," *IEEE Signal Process. Mag.*, vol. 30, no. 2, pp. 18–28, Mar. 2013.

[55] Y. Rane, S. C. D. Wang, and P. Ishwar, "Secure biometrics: Concepts, authentication architectures, and challenges," *IEEE Signal Process. Mag.*, vol. 30, no. 5, pp. 51–64, Sep. 2013.

[56] N. K. Ratha, S. Chikkerur, J. H. Connell, and R. M. Bolle, "Generating cancelable fingerprint templates," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 4, pp. 561–572, Apr. 2007.

[57] C. Rathgeb and A. Uhl, "A survey on biometric cryptosystems and cancelable biometrics," *EURASIP J. Inf. Secur.*, pp. 1–25, 2011.

[58] J. S. Ross and A. K. Jain, "From templates to images: Reconstructing fingerprints from minutiae points," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 4, pp. 544–560, Apr. 2007.

[59] E. A. Rua, E. Maiorana, J. L. A. Castro, and P. Campisi, "Biometric template protection using universal background models: An application to online signature," *IEEE Trans. Inf. Forensics Secur.*, vol. 7, no. 1, pp. 269–282, Feb. 2012.

[60] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[61] W. Sheng, G. Howells, M. C. Fairhurst, and F. Deravi, "Templatefree biometric key generation by means of fuzzy genetic clustering," *IEEE Trans. Inf. Forensics Secur.*, vol. 3, no. 2, pp. 183–192, Jun. 2008.

[62] W. Sheng, G. Howells, M. Fairhurst, F. Deravi, and S. Y. Chen, "Reliable and secure encryption key generation from fingerprints," *Inf. Manage. Comput. Secur.*, vol. 20, no. 3, pp. 207–221, 2012.

[63] Y. Sutcu, Q. Li, and N. Memon, "Protecting biometric templates with sketch: Theory and practice," *IEEE Trans. Inf. Forensics Secur.*, vol. 2, no. 3, pp. 503–512, Sep. 2007.

[64] A. B. J. Teoh and L. Y. Chong, "Secure speech template protection in speaker verification system," *Speech Commun.*, vol. 52, no. 2, pp. 150–163, 2010.

[65] A. B. J. Teoh, D. C. L. Ngo, and A. Goh, "Personalised cryptographic key generation based on face hashing," *Comput. Secur.*, vol. 23, no. 7, pp. 606–614, 2004.

[66] A. B. J. Teoh, W. K. Yip, and S. Lee, "Cancellable biometrics and annotations on BioHash," *Pattern Recognit.*, vol. 41, no. 6, pp. 2034–2044, 2008.

[67] P. Tuyls*et al.*, "Practical biometric authentication with template protection," in *Audio- and Video-Based Biometric Person Authentication*. Lecture Notes in Computer Science, vol. 3546. Berlin, Germany: Springer, 2005, pp. 436–446.

[68] E. Verbitskiy, P. Tuyls, D. Denteneer, and J. P. Linnartz, "Reliable biometric authentication with privacy protection," in *Proc. Benelux Symp. Inf. Theory*, Veldhoven, The Netherlands, 2003, pp. 125–132.

[69] C. Vielhauer, *Biometric User Authentication for IT Security—From Fundamentals to Handwriting*. Boston, MA, USA: Springer, 2006.

[70] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedel, "Constrained K-means clustering with background knowledge," in *Proc. 18th Int. Conf. Mach. Learn.*, San Francisco, CA, USA, 2001, pp. 577–584.

[71] J. Wayman, A. K. Jain, D. Maltoni, and D. Maio, *Biometric Systems: Technology, Design and Performance Evaluation*. London, U.K.: Springer, 2005.

[72] S. Xiong, J. Azimi, and X. Z. Fern, "Active learning of constraints for semi-supervised clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 43–54, Jan. 2014.

[73] Y. Yamazaki and N. Komatsu, "A secure communication system using biometric identity verification," *IEICE Trans. Inf. Syst.*, vol. E84-D, no. 7, pp. 879–884, Jul. 2001.

[74] W. K. Yip, A. Goh, D. C. L. Ngo, and A. B. J. Teoh, "Generation of replaceable cryptographic keys from dynamic handwritten signatures," in *Advances in Biometrics*. Lecture Notes in Computer Science, vol. 3832. Berlin, Germany: Springer, 2006, pp. 509–515.

[75] H. Zeng and Y. M. Cheung, "Semi-supervised maximum margin clustering with pairwise constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 926–939, May 2012.

[76] X. Zhu and A. B. Goldberg, *Introduction to Semi-Supervised Learning*. San Rafael, CA, USA: Morgan and Claypool, 2009.