

segments and/or the combination of segments that is most likely to indicate that a segment is not valid and so potential counterfeit activity.

[0075] Considering an $R_{\max} \times C_{\max}$ searching space where R_{\max} and C_{\max} are preset maximum numbers of rows and columns of the segmentation, the total number of possible ways to segment and combine the individual regions would be:

$$P = \sum_{i=1}^{R_{\max}} \sum_{j=1}^{C_{\max}} \binom{k}{ij}$$

[0076] where

$$\binom{k}{ij}$$

[0077] is the number of combinations of ij regions taken k at a time. If $R_{\max}=15$ and $C_{\max}=15$, $P=5.3923 \times 10^{67}$. This value exponentially increases following the changes of bigger R_{\max} and C_{\max} . Therefore, it is impossible to carry out exhaustive searching experiments. To achieve a useful result an optimized note segmentation and classifier combination strategy is proposed to gain the best trade-off between FN and FP performance can be achieved. In this, an optimal number of segments and an optimized classifier combination set are identified by a stochastic genetic algorithm (GA). This is useful because, as will be appreciated, different currencies have different feature distributions, so it is impossible to define a uniform segmentation and combination that is suitable for all possible currencies.

[0078] To optimize the note segmentations as well as the classifier combination at the same time, a stochastic optimization algorithm is used, in particular a genetic algorithm. This has one chromosome c , which is composed of three gene groups c_r , c_c and c_d , where $c_r=[r_1, r_2, \dots, r_{n_1}]$, $c_c=[c_1, c_2, \dots, c_{n_2}]$ and $c_d=[d_1, d_2, \dots, d_{n_3}]$ are all binary strings. The first two gene groups c_r and c_c respectively represent the number of rows and columns for a note segmentation. Their relationships with R and C are given by: $R=\text{BinToDec}(c_r)$ and $C=\text{BinToDec}(c_c)$, where "BinToDec" means transforming a number from a binary expression into decimal expression. Given the number of maximum rows R_{\max} and maximum columns C_{\max} which define the size of the intended segmentation search space, c_r 's length n_1 equals the length of the binary string expression of R_{\max} ; and c_c 's length n_2 equals the length of the binary string expression of C_{\max} . The third gene group c_d represents the combination of classifiers. Its elements d_i and length n_3 are defined as $d_i=0$ if $D_i \notin S$ and $d_i=1$ if $D_i \in S$ where $i=1, 2, \dots, n_3$; $n_3=R_{\max} \times C_{\max}$ and S is the optimized classifier combination set. For example, given $R_{\max}=3$ and $C_{\max}=3$, then n_1, n_2 and n_3 would respectively equal to 2, 2 and 9, and a chromosome

$$c=[c_r; c_c; c_d]=[11; 10; 110101000]$$

[0079] means segmenting a note into 3×2 (i.e. $R=3, C=2$) and combining the classifiers D_1, D_2, D_4 and D_6 respectively built on the 1st, 2nd, 4th and 6th sub-regions.

[0080] The GA for optimizing the system can be summarized as follows: (1) Initialize a random population of n chromosomes; (2) Perform crossover and mutation operations to create another n offspring; (3) carry out the validation operation, and calculate a fitness for each chromosome; (4) Select n chromosomes with the best fitness among all the parents and offspring as the next generation, and (5) if convergent or over the preset maximum iteration steps then stop, otherwise go to step (2). To implement this overall strategy, the GA has three gene operations for the chromosome. The first operation is gene crossover. Considering the characteristics of the chromosomes, two crossover points in one chromosome are set to complete the crossover operation with other chromosomes. One is the point between c_r and c_c ; and the other is the point in the middle of c_d . For example, if there are two chromosomes c_1 and c_2 represented as follows:

$$c_1=[c_r^1; c_c^1; d_1^1 d_2^1 \dots d_{n_3}^1]$$

$$c_2=[c_r^2; c_c^2; d_1^2 d_2^2 \dots d_{n_3}^2]$$

[0081] after crossover, two new chromosomes C_1^{new} and C_2^{new} are produced, where

$$c_1^{\text{new}}=[c_r^1; c_c^2; d_1^1 \dots d_{\text{INT}(n_3/2)}^1; d_{\text{INT}(n_3/2)+1}^2 \dots d_{n_3}^2]$$

$$c_2^{\text{new}}=[c_r^2; c_c^1; d_1^2 \dots d_{\text{INT}(n_3/2)}^2; d_{\text{INT}(n_3/2)+1}^1 \dots d_{n_3}^1]$$

[0082] The second GA operation is gene mutation. For this, a standard gene mutation operation is used, i.e. every time randomly selecting a gene (bit) to mutate from 1 to 0 or from 0 to 1. The third GA operation is validation. Because once the c_r and c_c have changed, only the first RC genes in c_d are effective, it is necessary to clean up the ineffective genes in order to avoid them affecting other chromosomes in further operations. Therefore the validation operation is designed to set all the n_3 -RC ineffective genes in c_d to be zeros.

[0083] Once the validation operation is carried out, a fitness function is calculated by balancing the performance of FN and FP in the validation set with the GA. Given N_1 genuine samples $G=\{g_{k1}\}$, where $k_1=1, \dots, N_1$ and N_2 counterfeit samples $F=\{f_{k2}\}$, where $k_2=1, \dots, N_2$ where both g_{k1} and f_{k2} are p -dimensional column vectors, then the decisions $y_i(g_{k1})$ and $y_i(f_{k2})$ of the i th classifier D_i towards validation samples g_{k1} and f_{k2} can be calculated. These are used to determine a fitness function, which is defined as:

$$f = \left(\frac{1}{N_1} \sum_{k_1=1}^{N_1} \prod_{(i: D_i \in S)} y_i(g_{k1}) \right)^2 + \left(\frac{1}{N_2} \left(N_2 - \sum_{k_2=1}^{N_2} \prod_{(i: D_i \in S)} y_i(f_{k2}) \right) \right)^2$$

[0084] where the first term indicates the True Negative (TN: =1-FP) performance and the second term indicates the True Positive (TP: =1-FN) performance. A chromosome that has a larger value of fitness function will have better performance when applying the segmentation and combination that this chromosome represents.

[0085] An alternative approach to evaluate the fitness of chromosomes is to use Receiver Operating Characteristics (ROC). This is described in the paper "Genetic programming for combining classifiers" by Langdon et al, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO' 2001), San Francisco, USA, Morgan