Defence Science Journal, Vol. 67, No. 1, January 2017, pp. 59-65, DOI : 10.14429/dsj.67.9153

Classification of Encryption Algorithms using Fisher's Discriminant Analysis

Prabhat Kumar Ray[*], Shri Kant[#], Bimal K. Roy[@], and Ayanendranath Basu[!]

[*]*Home (C & E) Department, Kolkata -700 001, India*
[#]*RTDC Sharda University, Greater Noida - 201 306, India*
[@]*Applied Statistics Unit, Indian Statistical Institute, Kolkata - 700 108, India*
[!]*ISRU, Indian Statistical Institute, Kolkata - 700 108, India*
[*]*E-mail: prabhat.k.ray@gmail.com*

ABSTRACT

Fisher's discriminant analysis (FDA) is a method used in statistics and machine learning which can often lead to good classification between several populations by maximising the separation between the populations. There is, literally, a huge body of literature which deals with the application of Fisher's discriminant analysis in problems of statistical classification and describes its desirable properties. The method is supposed to work well whenever there is a suitable numerical feature vector having different statistical distributions under the different classes. In this paper we will present some applications of FDA that discriminate between cipher texts in terms of a finite set of encryption algorithms. Specifically, we use ten algorithms, five each of stream and block cipher types, and, given a cipher text, try to identify which algorithm was used in the encryption of the given cipher. Our results display good classification with some of the features. Our limited study clearly shows that further exploration of this classification problem based on FDA could be worthwhile.

Keywords: Fisher's discriminant analysis; Encryption; Classification

1. INTRODUCTION

Methods to classify encryption algorithms have enormous applications. Fisher's Discriminant Analysis (FDA) has been used in the areas of image processing and pattern recognition[6], machine intelligence[2]and in the classification of speech/ music[1]. However, we have not found the application of FDA in classifying cipher texts, although the Hidden Markov Model has been used in a similar context by Ray[7], *et al.* and other pattern recognition techniques has been used by Sharif[9], *et al.* We consider the application of FDA to classify cipher texts; the method is applied for detecting the algorithm used for the encryption of some meaningful plain text. It is essential for a cryptanalyst to predict the encryption method so that the appropriate attack may be chosen. Here we aim to predict the algorithm used to construct a particular ciphertext. The results indicate that the FDA method has reasonable classification properties.

2. THE BASIC DATABASE

We have created a database of 200 files each of sizes 1 kb, 2 kb, 5 kb, and 10 kb. All the source (plaintext) files represent meaningful English texts. Given a plaintext file, we convert each character of the plaintext file to the corresponding ASCII value, which is represented in binary format through an appropriate coding scheme. This produces the binary (as opposed to English) plaintext file, the input of the encryption algorithms. We have used ten algorithms for encryption, five each of the stream cipher and block cipher types. Encryption algorithms are briefly described.

Received : 30 August 2016, Revised : 25 October 2016
Accepted : 02 December 2016, Online published : 22 December 2016

3. CIPHER ALGORITHMS

Symmetric cryptosystems are divided into two types: stream cipher and block cipher. In stream ciphers, plaintexts are combined with a pseudorandom cipher bit stream (key stream) by an exclusive-or (XOR) operation. Each bit x_i is encrypted by adding a secret key secret bit s_i(modulo 2). Block ciphers involve an enciphering transformation on each "message block" independently. A block cipher breaks the plaintext P = (p_1, p_2,…,p_n) into several message blocks with the same length and transforms it to the cipher text $C = (c_1, c_2,\ldots, c_n)$ via an encryption function controlled by a secret key K. The first four of our five stream ciphers are based on the linear feedback shift register (LFSR) which is a mechanism for generating a sequence of binary bits. At each clocking instant, the contents are shifted right by one position, and the XOR of a subset of the cell contents is placed in the left most cell. Let the binary numbers $s_0, s_1, s_2..s_{k-1}$ represent the initial state of the register, and suppose that we have a fixed sequence of binary evolutions $a_0, a_1, a_2,\ldots, a_{k-1}$. Registers are composed of bits numbered from right to left, i.e., s_0 is the right most bit of the block. At each step the contents of the cells of the register are shifted right by one position, and the right most bit of the

register is the output. The function value $f = \sum_{i=0}^{k-1} a_i s_i$ (modulo 2) is placed in the leftmost cell. For our first cipher (stream1) we have used an LFSR with register length 64, but the output keystream has been generated by a nonlinear filter generator function g based on the register sequence $\{s_i\}$. Given the sequence $\{s_{63}, s_{62},\ldots,s_0\}$, the output at the current stage will be a nonlinear function of the $\{s_i\}$'s, rather than the output stream of the LFSR. We have chosen the sequence $\{\boldsymbol{a}_i\}$ according to the coefficients of a 64 degree primitive polynomial on GF(2), the Galois field of two elements; the coefficients are either 0 or 1.For stream cipher 2(stream2) the method is the same as the previous one; the only difference is that the possible values in the register $\{s_i\}$ and the evolution $\{\boldsymbol{a}_i\}$ are 0, 1, α and 1 + α, where α is a root of the equation $x^2 + x + 1 = 0$. In the two bit representation 0 represent 00, 1 represent 01, α represent 10 and 1 + α represent 11. For Stream Cipher 3 (stream3) we have taken three different LFSRs of length 56, 64 and 72, respectively. The fixed sequences of $\{a_i\}$'sfor these three LFSRs are the coefficients of a 56 degree, a 64 degree, and a 72 degree primitive polynomial respectively. These three LFSRs will produce three output streams. Let these three streams be $X_n^1, X_n^2, X_n^{13}, n = 1, 2, \ldots$. We will input these three streams in a function $f: \{0, 1\}^3 \rightarrow \{0, 1\}$ and get a specific output based upon the input stream. For this we choose the multiplexer function defined as

$$f(X_n^1, X_n^2, X_n^3) = \begin{cases} X_n^1 & \text{if } X_n^3 = 0 \\ X_n^2 & \text{if } X_n^3 = 1 \end{cases}$$

Hence we can get the keystream output as $\{Y_1, Y_2, \ldots\}$ where $Y_n = f(X_n^1, X_n^2, X_n^3)$.

In stream cipher 4 (stream 4), we have taken the three different LFSRs used in stream 3. The fixed sequences $\{\boldsymbol{a}_i\}$ of these three LFSRs are, respectively, the coefficients of 56, 64 and 72 degree primitive polynomial. The three LFSRs will produce output streamsX_n^1, X_n^2, X_n^3, n = 1, 2,… We will input these streams with the output of the previous cycle in a function f: $\{0, 1\}^4 \rightarrow \{0, 1\}$ and get an output depending upon the input stream. The function chosen by us can be defined as:

$$f(X_n^1, X_n^2, X_n^3, Y_{n-1}) = \begin{cases} X_n^1 \text{ if } X_n^3 = 0, Y_{n-1} = 0 \\ \overline{X}_n^1 \text{ if } X_n^3 = 0, Y_{n-1} = 1 \\ X_n^2 \text{ if } X_n^3 = 0, Y_{n-1} = 0 \\ \overline{X}_n^2 \text{ if } X_n^3 = 1, Y_{n-1} = 1 \end{cases}$$

Then our keystream is Y_0, Y_1, Y_2…..where Y_0= 0 by convention. Here $\overline{x} = |1 - x|$

Stream Cipher 5(stream5) in our discussion is the variably modified permutation composition (VMPC) cipher based on the VMPC function. Apart from the VMPC function, the cipher employs two other operations: they are updates of an internal 8 bit variable(s) and a swap operation of the internal permutation (P). The key scheduling algorithm (KSA) of VMPC transforms a cryptographic key from 128 to 512 bits and an Initialisation Vector into a 256-element internal permutation (P). The VMPC generates the output in bytes (8 bits), rather than in single bit. Let P and Q be n-element permutations, representing one-to-one mappings $A \rightarrow A$, where $A = \{0, 1, \ldots, n - 1\}$. Thus the dimension of the P, Q arrays equals n. Let k (< n) represent the level of the VMPC function and '+' represent addition modulo n. A k-level VMPC functionVMPC_k is a transformation of P into Q, where $Q[x] = P[P_k[P_{k-1}[\ldots[P_1[P[x]]]\ldots]]]$, $x \in \{0, 1, \ldots, n-1\}$. Here P_i is an n-element permutation such that $P_i[x] = fi(P[x])$, where f_i is any function satisfying $P_i[x] \neq P[x] \neq P_j[x]$ *for* $i \in \{1, \ldots, k\}, j \in \{1 \ldots k\}/\{i\}$. For simplicity, f_i is assumed to be $f_i(x) = x + i$.(This function is used in the basic reference to VMPC above). We will only implement the level 1 VMPC, $Q = \text{VMPC}(P)$ will be used interchangeably with $Q = \text{VMPC}_1(P)$. See Zoltak[10], for a useful description of the VMPC cipher.

Our first two block ciphers (block1 and block2) are Feistel ciphers or DES (Data Encryption Standard)-like ciphers. DES is a 16-round 64 bit Feistel Cipher. The algorithm is designed to encipher and decipher blocks of data consisting of 64 bits under control of a 64-bit key. A block to be enciphered is subjected to an initial permutation *IP*, then to a complex key-dependent computation and finally to a permutation IP^{-1} which is the inverse of the initial permutation. The key-dependent computation can be simply defined in terms of a function *f*, called the cipher function, and a function KS, called the key schedule. Initial permutation IP is a given table by which initial 64-bit should be rearranged and split into two half of 32-bits each. Let the two halves be initially referred to as L and R. Then, after the first iteration, we have

$L' = R;$

$R' = L \oplus f(R; K);$

where $\oplus$ denotes bit-by-bit addition modulo 2 and K is a block of 48 bits chosen from the 64-bit keys. For calculation of cipher function *f* at any cycle, the right block R (32 bits) is subjected to a function *E*which takes this 32 bit input and yields a 48 bit output block. The selection of bits (of R) in the E function is performed according to anorder Table. The output of the E function is XORed with the subkey*K* for that cycle. The resultant 48 bits are partitioned into eight contiguous six bit blocks, *B1,…..,B8*; formally, *B1 B2…..B8 = E(R)* $\oplus$ *K*. Each block B_i is the input to a selection function (S-box) S_i, *i = 1, 2,…..,6*. Each selection function takes 6 bit inputs and produces4 bit outputs. Then the 8 blocks $S_1(B_1), S_2(B_2), \ldots .., S_8(B_8)$ are consolidated into $S_1(B_1)S_2(B_2)\ldots..S_8(B_8)$ through concatenation, and a single 32 bit block is obtained. In key schedule algorithms a 64 bit key is arranged through two permuted choices to obtain a 48 bit key. At each round the key block is obtained from the previous one by a left shift.

S-Box:

Due to the similarity in the nature of the *S* -boxes, a proper definition of S_1 is sufficient for understanding the *S*-box structure. If *B* is a block of 6 bits, then $S_1(B)$ is determined from Table 1 as follow, The first and last bits of *B* represent in base 2 a number *i*in the range 0 to 3. The middle 4 bits of *B* represent in base 2 a number *j*in the range 0 to 15. We look up the required number in the table in the i^{th} row and j^{th} column. It is a number in the range 0 to 15 and is uniquely represented by a 4 bit block. That block is the output $S_1(B)$ of S_1 for the input *B*. For example, for input 011011 the row is 1 and the

Table 1. S-Box

Col No.-> Row No.↓	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

column is 13. The number that appears in row 1 and column 13 equals 5 so that the output is 0101. Block1 differs from DES in that instead of choosing 8 different *S* -boxes, we choose 8 identical selection functions, all equal to S_1. This method will result in a Feistel Cipher (32 bit + 32 bit; two blocks) with a Feistel function simpler than DES. The ciphertext can be generated as in DES by supplying a secret key. In block2 we have slightly changed the cipher function *f* of DES. Here after passing through all the S-boxes the resultant 32-bit blocks are rearranged according to a permutation table.

Block ciphers 3 and 4 (block3 and block4) are based on the Substitution Permutation Network (SPN) . An R-round SPN requires (R + 1) N-bit subkeys, k^1,k^2 ,…..,k^{R+1}. A round consists of three stages. The first stage is the keymixingstage, the N-bit round input is bitwise XORed with the round subkey. The second stage is the substitution stage, the resulting block is partitioned into M subblocks of size n (N = Mn), and each subblock becomes the input to a bijective$n \times n$substitution box (S-box)-a bijective mapping from $\{0, 1\}^n$ to $\{0, 1\}^n$. In the final stage, the output from the substitution stage is processed through an invertible N-bit linear transformation. A final subkey, k^{R+1}, is XORed with the output of round R to form the ciphertext. Decryption is accomplished by running the SPN backwards. Subkey k^{R+1} is XORed with the ciphertext, and in each round *r* (from R to 1), the inverse linear transformation is applied, followed by the inverse *S*-boxes. For blocks 3 and 4 we have chosen N = 64, M = 8, n = 8 and R = 6. The only difference between them is that in block 3 we have used the S-Box as a mapping function a linear function but in case of block 4 the non-linear mapping function (*S*-Box) is used instead of linear function.

The final block cipher algorithm of our discussion (block5) is MARS. MARS is a shared-key block cipher, with a block size of 128 bits and a variable key size, ranging from 128 to over 400 bits. MARS takes as input (and produces as output) four 32-bit data words. The process of MARS consists of three stages: forward mixing, cryptographic core and backward mixing.

The first phase provides rapid mixing and key avalanche, to frustrate chosen-plaintext attacks, and to make it harder to strip out rounds of the cryptographic core in linear and differential attacks. It consists of addition of key words to the data words, followed by eight rounds of S-box based, unkeyed type-3 Feistel mixing (in 'forward mode').The second phase is the 'cryptographic core' of the cipher, consisting of sixteen rounds of keyed type-3 Feistel transformations. We perform the first eight rounds in 'forward mode' while the last eight rounds in 'backwards mode'. The last phase again provides rapid mixing and key avalanche, to protect against chosen-cipher text attacks. This phase is essentially the inverse of the first phase, consisting of eight rounds of the same type-3 Feistel mixing as in the first phase (except in 'backwards mode'), followed by subtraction of key words from the data words.

4. FISHER'S DISCRIMINANT ANALYSIS

We provide a discussion of Fisher's Discriminant Analysis following Johnson and Wichern[5]. Suppose have *g* populations. There is a variable of interest *X*, using which we want to discriminate between the populations for future test data. In FDA, it is not necessary to assume that the populations are multivariate normal. However, we assume that the population covariance matrices are equal and nonsingular. If the covariance matrix of *X* in the i^{th} population is Σ_i, then $\Sigma_1 = \Sigma_2 = \cdots = \Sigma_g = \Sigma$, where $det(\Sigma) > 0$ and Σ is a $p \times p$ matrix. Let the mean of *X* in the i^{th} population be denoted by μ_i, i=1,2,…..,g. Let $\bar{\mu} = \frac{1}{g}\sum_{i=1}^{g}\mu_i$ be the grand mean of *X*. Let B_0 be the between group sum of squares and products given by $B_0 = \sum_{i=0}^{g}(\mu_i - \bar{\mu})(\mu_i - \bar{\mu})^T$. We consider linear combinations $Y = l^T X$ where *l* is some appropriate $p \times l$ vector, to transform *X* to a scalar . The mean of *Y* in the i^{th} population is $E(Y) = l^T E(X \mid \mu_i) = l^T \mu_i$ and the variance of *Y* in the i^{th} population is a constant (which does not depend on *i*) and equals $Var(Y) = l^T Cov(X) l = l^T \Sigma l$. Let $\mu_{iY} = l^T \mu_i$ be the mean of *Y* in the i^{th} population. Then the overall mean of *Y* over all the populations is $\bar{\mu}_Y = \frac{1}{g}\sum_{i=1}^{g}\mu_{iY} = \frac{1}{g}\sum_{i=1}^{g} l^T\mu_i = l^T\left(\frac{1}{g}\sum_{i=1}^{g}\mu_i\right) = l^T\bar{\mu}$. We want to choose the vector *l*, so as to maximise the separation between the populations with respect to the transformed variable *Y*. We will measure this variation with the following quantity

$$\left(\frac{Sum\ of\ squared\ distance\ from\ Population\ to\ overall\ mean\ of\ Y}{(Variance\ of\ Y)}\right) = \frac{\sum_{i=1}^{g}\left(l^T\mu_i - l^T\bar{\mu}\right)^2}{\sigma_Y^2}$$

$$= \frac{\sum_{i=1}^{g}(\mu_i - \bar{\mu})^2}{\sigma_Y^2} = \frac{l^T\left(\sum_{i=1}^{g}(\mu_i - \bar{\mu})(\mu_i - \bar{\mu})^T\right)l}{l^T\Sigma l}$$

Thus $\frac{\sum_{i=1}^{g}\left(l^T\mu_i - l^T\bar{\mu}\right)^2}{\sigma_Y^2} = \frac{l^T B_0 l}{l^T \sum l}$, and to choose *l* so as to maximise the separation between the populations, one has to find the value of *l* which maximises the right hand side of the above equation. However, normally μ_i's and Σ will be unknown and in practice we will use estimates from the sample. Let n_i be the sample size of i^{th} group. Thus we will use $\bar{x}_i = \frac{1}{n_i}\sum_{j=1}^{n_i} X_{ij}$ as an estimate of μ_i, and $\bar{X} = \frac{\sum_{i=1}^{g} n_i \bar{X}_i}{\sum_{i=1}^{g} n_i} = \frac{\sum_{i=1}^{g}\sum_{j=1}^{n_j} X_{ij}}{\sum_{i=1}^{g} n_i}$ as an

estimate of $\bar{\mu}$, and $S_{Pooled} = W/(n_1 + n_2 + n_3 + \ldots\ldots + n_g - g)$, as an estimate of Σ where $W = \sum_{i=1}^{g}\sum_{j=1}^{n_j}\left(X_{ij} - \bar{X}_i\right)\left(X_{ij} - \bar{X}_i\right)^T$. Ignoring the constants, we can therefore maximise the separation between the populations by maximising the measure $MS = \frac{l^T \hat{B}_0 l}{l^T W l}$, over l, where $\hat{B}_0 = \sum_{i=1}^{g}\left(\bar{X}_i - \bar{X}\right)\left(\bar{X}_i - \bar{X}_i\right)^T$.

Such maximisations are standard problem in linear algebra and the quantity MS is maximised by $l = e_i$, where e_i is the eigenvector corresponding to the largest eigenvalue of the matrix $W^{-1}\hat{B}_0$

Let $e_1, e_2, \ldots, e_s$ be the eigenvectors corresponding to the nonzero eigenvalues $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_s$ of $W^{-1}\hat{B}_0$ Note that $s \leq \min(g-1, p)$ We will call the quantity $e_1^T X$ the *1st* discrimination function. Notice that e_1 maximises the measure MS. The vector $l = e_2$ maximises the measure MS subject to $e_1^T W l = 0$ i.e. subject to the constraints that the first two discriminants are uncorrelated. Similarly, the vector $e_j, j = 1, 2, \ldots, s$ maximises the measure MS subject to $e_i^T W l = 0, i = 1, 2, \ldots j-1$. The j^{th} discrinionant is uncorrelated with all the previous ones.

The discrimination proceeds as follows: Let $Y = (l_1^T X, l_2^T X, \ldots . l_s^T X)$ Using training data, one computes the mean of *Y* in that population. For a new test case, one first computes the vector *Y* of discriminants, and then determines the Euclidean distance of this vector from each of the population means of *Y*. The case is assigned to the population that minimises this distance.

5. F-RATIO

Sometimes the lengths of the feature vectors are too large, and it becomes unmanageable to use the entire feature in the analysis. In such cases we can reduce the length of the vector by retaining only those components which have the greatest discriminating capacity. This is done by computing the F-ratio of between groups and within groups sum of squares, and choosing those components with the largest F-ratios. The F-ratio is calculated as follows: let the given data for a particular component be denoted by x_{ij}, the observation on the j^{th} case of for the i^{th} population. Then the F-ratio is calculated as $F = \frac{\sum_{i=1}^{g}\left(\bar{x}_i - \bar{x}_{...}\right)^2}{\sum_{i=1}^{g}\sum_{j=1}^{n}\left(x_{ij} - \bar{x}_{i....}\right)^2}$, $\bar{x}_i = \frac{\sum^{n}{}_{j=1}\sum_{j=1}^{n_j} x_{ij}}{n}$, $\bar{x}_{i..} = \frac{\sum_{i=1}^{k}\sum_{j=1}^{n} x_{ij}}{ng}$. Here *n* is the number of cases for each population, and *g* is the number of populations. The F-ratio uses the same philosophy and measure as in Section 4. One determines the separation between the populations for each scalar variable. We have retained the top five components.

6. DISCRIMINATION BASED ON AUTOCORRELATION

Given a finite set of time series data $\{x_1, x_2, \ldots, x_n\}$ the autocorrelation coefficient at lag *h* measures the correlation between observations *h* time units apart. The autocorrelation coefficient $\rho(h)$ at lag *h* is formally defined as:

$$\rho(h) = \frac{\sum_{t=1}^{n-h}(x_{t+h} - \bar{x})(x_t - \bar{x})}{\sum_{t=1}^{n}(x_t - \bar{x})^2} \text{ where } \bar{x} = \sum_{i=1}^{n} x_i .$$

The autocorrelation coefficients always lie between -1 and 1 and a greater magnitude of the autocorrelation coefficient at lag *h* indicates greater amount of linear dependency between data shifted at lag *h*. Our basic input is the raw sequence of 0's and 1's of the cipher text. We will use features based on autocorrelation and discrete Fourier transform of the cipher text data. We classify cipher text, which have been encrypted by any of the ten different algorithms. For training we have chosen 200 files of sizes 1k, 2k and 5k each from each of the 10 different algorithms. Thus there are 2000 training files of each size.

7. FEATURES BASED ON AUTOCORRELATION

7.1 Feature based on Raw Cipher

We take the raw sequence of '0's and '1's from each cipher file and construct the autocorrelations of lags up to 20 over all the cipher files. This produces a matrix of autocorrelations of dimension 2000×20. We then construct the F-ratios corresponding to each lag, and choose the 5 lags giving the highest autocorrelation and get a 2000×5 matrix. We treat this as a sample of 2000 five dimensional observations with known class memberships, and perform FDA on this data.

7.2 Feature based on Proportions of '1' in 100 Bits

Here we take the raw sequence as in previous feature then choose segments of 100 bits (sequence of '0's and '1's) from each cipher files, with 50 bits of overlap between the successive segments. For each segment, we compute the frequency of 1's, and resulting sequence of frequencies is now our feature data. We calculate the autocorrelation from those data, find the five best discriminating lags and perform Fisher's discrimination as above.

7.3 Feature based on Length of the Largest Run in 100 Bits

We again choose segments of 100 bits. In each segment we choose the feature to be the length of the largest run of 1's in this segment. This length is then divided by 100 (the length of the segment). We calculate the autocorrelations for the resulting sequence. We calculate the five best discriminating lags. Fisher's discrimination is then performed as in the previous cases.

7.4 Feature based on Number of Runs of '1' in 100 Bits

Again we choose segments of 100 bits. In each segment we choose the number of runs of 1. This number is then divided by 100 (the length of the segment). We calculate the autocorrelation from the resulting sequence. We calculate the five best discriminating lags and Fisher's discrimination is then performed as in the previous cases.

8. DISCRIMINANT ANALYSIS BASED ON THE DISCRETE FOURIER TRANSFORM

The discrete Fourier transform (DFT) is actually a transformation from the time domain to the frequency domain.

The procedure for constructing the Discrete Fourier Transform is available in many standard texts such as Cormen[4], *et al.* and a more practical method was given by Cooley and Tukey[3] called the Fast Fourier Transform. The only condition for implementing this process is that the series should be of a length which is a power of 2. If not then we pad the series with some zeros at the end to take it to the next power of 2. For DFT we use the same features corresponding to Sections 7.2, 7.3 and 7.4 as done in case of autocorrelation.

9. RESULTS

With the database as described in Section-2 we have found 2000 encrypted file form 200 file of same size and key type applying 10 encryption algorithm. We classify each of these 2000 files in terms of the source algorithms using the FDA technique. The results are in matrix form for particular features, size of files and key types. Tables 2-5 provide a selection of results generated by FDA for different such combinations. The five stream cipher algorithms are denoted as *S1 ,…,S5.*, and the five block cipher algorithms are denoted as *B1 ,…,B5.*The rows represent true membership and the columns represent predicted membership. The total of each row in any given table is 200 and the diagonals represent correct classification. If the classification is random, 200 files will be correctly classified on the average. We wish to show that FDA provides better than random classification. If the number of correct classifications appear to be significantly higher than 20 for the diagonal cells of the tables (or significantly higher than 200 for the sum of the diagonals) one could claim that the method is better than random allocation. While we provide the tests to determine whether the number of correct allocations is better than that of random allocation in the next section, the tables immediately draw our attention to the fact that FDA based on the original cipher is hugely successful in identifying ciphers encrypted by block2. Apparently the non-linear S-boxes impart a vulnerability in the ciphers which is exploited by FDA.

Table 2. No of files for test : 2000, trained with : key type (Single) –size (1k), tested with : key type (Single)–size(2k), feature : autocorrelation on original (raw) cipher

	s1	s2	s3	s4	s5	b1	b2	b3	b4	b5
s1	**90**	1	10	15	53	2	8	5	5	11
s2	12	**49**	22	20	24	9	38	13	3	10
s3	5	7	**85**	34	17	6	2	24	9	11
s4	8	36	31	**49**	18	19	7	11	9	12
s5	12	27	14	13	**48**	13	28	26	8	11
b1	19	17	14	15	24	**10**	39	19	28	15
b2	9	6	5	1	6	2	**138**	7	17	9
b3	25	14	20	21	16	17	24	**15**	32	16
b4	20	7	21	16	21	9	31	27	**29**	19
b5	16	13	19	25	12	18	38	17	2 4	**18**

Table 3. No of files for test : 2000, trained with : key type (Multiple) – size (1k), tested with : key type (Single)– size (2k), feature : autocorrelation on original (raw) cipher

	s1	s2	s3	s4	s5	b1	b2	b3	b4	b5
s1	**6**	125	23	6	12	0	18	2	7	1
s2	7	**30**	20	4	64	5	46	4	10	10
s3	26	24	**37**	16	44	4	2	2	3	42
s4	15	24	31	**11**	35	3	9	15	13	44
s5	6	69	31	11	**16**	1	32	21	6	7
b1	13	32	16	17	33	**6**	45	8	10	20
b2	2	12	3	5	20	2	**143**	5	5	3
b3	18	33	20	15	31	6	33	**10**	14	20
b4	12	45	23	13	34	5	38	9	**11**	10
b5	14	30	20	9	43	7	32	16	15	**14**

10. TESTS OF HYPOTHESIS

If the FDA based classification is better than random classification, one would expect that the sum of the diagonals of a table of the above type to be significantly higher than 200 on the average. To explore this, all tables of a particular feature were chosen, and the cases where the training files were not repeated as test files were considered. There are 30 such tables

Table 4. No of files for test : 2000, trained with : key type (Multiple) – size(5k), tested with : key type(Single)– size(5k), feature : autocorrelation on frequency 1 in 100

	s1	s2	s3	s4	s5	b1	b2	b3	b4	b5
s1	**15**	3	10	20	51	3	56	33	0	9
s2	25	**10**	2	9	57	1	69	19	0	8
s3	21	8	**6**	17	23	6	103	9	0	7
s4	33	3	5	**24**	76	0	21	23	0	15
s5	18	1	2	8	**124**	3	18	11	0	15
b1	25	4	9	24	57	**4**	43	21	1	12
b2	38	3	7	20	54	3	**41**	11	0	23
b3	33	1	5	29	43	6	44	**27**	1	11
b4	27	2	9	20	61	4	40	23	**0**	14
b5	32	4	5	27	56	1	42	20	0	**13**

Table 5. No of files for test : 2000, trained with : key type (Single) – size (1k), tested with : key type (Multiple)– size(1k), feature : autocorrelation on run 1 in 100

	s1	s2	s3	s4	s5	b1	b2	b3	b4	b5
s1	**40**	31	24	39	21	5	20	5	13	2
s2	47	**24**	24	30	30	5	22	1	16	1
s3	56	27	**16**	26	21	8	20	4	20	2
s4	44	41	11	**25**	26	7	20	7	17	2
s5	50	31	10	36	**28**	9	16	4	16	0
b1	48	27	19	30	28	**10**	17	4	15	2
b2	47	27	13	26	29	14	**27**	7	8	2
b3	34	26	26	27	30	10	16	**10**	15	6
b4	39	37	26	27	26	7	21	3	**13**	1
b5	40	33	16	32	31	9	20	5	12	**2**

for each feature. In each case, we looked at the distribution of the sum of the diagonals over the 30 tables, and did a test of hypothesis to determine whether the location parameter was greater than 200 using different parametric and non-parametric tests. Assuming that the sum of the diagonals has a a normal distribution with mean μ, the *t*-test for the hypothesis $H_0 : \mu = 200$ against the greater than alternative over the 30 tables led to *p*-values of 0, 0.0003 and 0.0010 for the FDA for autocorrelation based feature on (i) the raw cipher, (ii) the proportion of 1 in 100 bits and (iii) the length of the largest run in 100 bits, respectively. For nonparametric tests, even the least powerful sign test for the hypothesis $H_0 : \eta = 200$ against the greater than alternative produces p-values of 9.3×10^{-10}, 0.0026, and 0.0026, respectively, where η is the population median of the sum of the diagonals. In each of the above cases the observed statistics are highly significant, suggesting that the FDA based methods may be doing substantially better than random classification. While the above analysis involves only sums of the diagonals of the tables, one should check whether the general distribution of the observations along the cells of the tables follow an overall nonrandom pattern. Under randomness, each entry of the table is a random variable with mean 20. A test for this null hypothesis, $H_0 : \theta_{i,j} = 20$ is provided by the standard Pearson's chi-square statistic (with 90 degree of freedom), where θ_{ij} is the expected frequency in the ij^{th} cell. For Tables 2-5, these statistics are 1782.30, 2347.300, 2473.60 and 868.40 respectively, and in each case the p-value is practically zero. Clearly the overall assignment along cells is not random.

11. COMPARISON WITH HMM BASED AND OTHER CLASSIFICATIONS

We compare our results with the hidden Markov HMM (model) based classification results of Ray[7], *et al.* If there is a difference in FDA based results over the HMM based ones it would be expected that the differences of the sum of the diagonals of the confusion matrices would be different from zero for the same type of key and file size. To compare we have chosen 30 tables where the training and test files are not the same for each of common feature used (Proportion of 1 in 100 and Run of 1 in 100 bits for autocorrelation based on FDA and HMM based classification) and then taken the differences of their diagonal sums. We ran t-tests of the hypothesis $H_0 : \mu = 0$ against the not equal to alternative where μ is the mean of the differences of sums of diagonals (between the FDA and HMM based classification) over the 30 tables, under the assumption that the difference of the sum of diagonals is normally distributed. The p-values were 0.790037 and 0.2364 for Proportion of 1 in 100 bits and Run of 1 in 100 bits respectively. The sign test for $H_0 : \eta = 0$ against the not equal to alternative produces *p*-values of 0.855 and 0.3615, respectively, where η is the population median of differences of sums of diagonals. All the statistics suggest the lack of any significant difference. Neither the HMM or FDA based methods are necessarily better than the other.

It is relevant, in this connection to refer to the work done by Saxena[8] at the Indian Institute of Technology, Kanpur, which was brought to our attention by one of the referee. This work (and the references therein) has concentrated on the problem of classifying two sets of cipher text generated by Blowfish: RC4 (B_R) and Camellia: RC4 (C_R). This work has used test vectors of length 320 bits at a time, and attempted to optimise the performance. When a cipher text generated by a particular cipher is classified with more than 50 per cent accuracy it is referred to as trivially-good-test vector. In the process the linear programming optimisation technique is used. Later various positions (even, odd, random, etc.) of the test vectors are changed and the performance is analysed to determine whether they qualify as good test vectors. For classification various methods such as support vector machines are used.

Our experimental set up is entirely different from the work described in the previous paragraph. We have also considered a larger number of ciphers. A comparison of the two methods, however, could be meaningful. This will require an extensive study where both methods are compared under the same footing. We hope to take up this study in the future.

12. CONCLUSIONS AND FUTURE RESEARCH

In this study we used the FDA to classify encryption algorithms using different feature vectors. Further studies are necessary to establish the strength of the classification process, but the indications of this limited study are clearly encouraging. We strongly believe that this approach should be further explored; in addition, a full scale comparison of this method with those based on the group of techniques represented by Saxena[8] would also be worthwhile.

REFERENCES

1. Alexandre-Cortizo, E.; Rosa-Zurera, M. & Lopez-Ferreras, F. Application of Fisher linear discriminant analysis to speech/music classification. *In* Computer as a Tool, 2005. EUROCON 2005. The International Conference on IEEE, 2005, **2**, 1666–1669. doi: 10. 09/EURCON.2005.1630291
2. Cooke, T. Two variations on Fisher's linear discriminant for pattern recognition. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 2002, **24**(2), 268–273. doi: 10.1109/34.982904
3. Cooley, J.W. & Tukey, J.W. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.*, 1965, **19**, 297–301. doi: 10.2307/2003354
4. Cormen, T.H.; Leiserson, C.E.; Stein, C & Ronald, L. Rivest. Introduction to algorithms. 3rd Ed. MIT press and McGrawHill, 2009.
5. Johnson, R.A. & Wichern, D.W. Applied multivariate statistical analysis. 6th Ed., Prentice Hall, New Jersey, 2007.
6. Liang, Z.; Shi, P & Zhang, D. Two-dimensional Fisher Discriminant Analysis and its application to face recognition. *In* Computer Vision–P. J. Narayanan *et al*. (Eds)-ACCV, Springer, 2006, 130–139.
7. Ray, P.K.; Shri Kant; Roy, B.K. & Basu, A. Classification

of encryption algorithms using the Hidden Markov Model. *Calcutta Statistical Association Bulletin*, 2012, **64**, 277–290.
doi: 10.1177/0008068320120309.

8. Saxena, Gaurab. Classification of ciphers using machine learning. Department of Computer Science and Engineering, IIT Kanpur, 2008. Master's Thesis.
9. Sharif, S. O.; Kuncheva, L. & Mansoo, S. Classifying encryption algorithms using pattern recognition techniques. *In* Information Theory and Information Security (ICITIS), IEEE International Conference, 2010, pp. 1168–117.
doi: 10.1109/ICITIS.2010.5689769
10. Zoltak, B. VMPC one-way function and stream cipher. *In* Lecture Notes in Computer Science, *Springer*, 2004, **3017**, 210–225.

CONTRIBUTORS

Mr Prabhat K. Ray obtained MCA from Visva-Bharati University, Santiniketan, India. Presently, he is a System Manager at the Home (C&E) Department, Government of West Bengal, India.
In the current study, his contributions include: Partial analysis of the problem, and necessary coding implementing the classification method.

Dr Shri Kant received his PhD (Mathematics) from Banaras Hindu University. He is currently working as Professor and Dean Research at Sharda University. Prior to this he has served DRDO as Coordinator and Director, Joint Cipher Bureau, DRDO, Matcalfe House, Delhi.
In the current study, he proposed the need for a statistical classification of cipher texts within this research group and worked at each stage of the implementation of the statistical techniques to assess their cryptanalytic feasibility, and also monitored the implementation of the coding work.

Prof. Bimal Roy received his PhD (Combinatorics) from the University of Waterloo, Canada. He is currently working as a Professor at the Applied Statistics Unit of the Indian Statistical Institute.
In the current study, he conceptualised the nature of the study and designed the form of the experiment, including the choice of all the ciphers (the five stream cipher and the five block ciphers). He also provided partial input for the statistical part of the work.

Prof. Ayanendranath Basu got his PhD (Statistics) from the Pennsylvania State University, USA. He is currently working as a Professor at the Interdisciplinary Statistical Research Unit of the Indian Statistical Institute.
In the current study, he provided most of the statistical input for the paper, including the plan for the use of Fisher's Discriminant Analysis and most of the statistical tests done in the paper.