An ECG Data Compression Method Via Local Maxima and ASCII Character Encoding

Raj Kumar, Sandeep Kumar, Manish Rai, Sanket Kumar

Abstract— The electrocardiogram (ECG) compression method presented in this paper is based on ASCII character encoding. In this compression methodology, at first individual standard deviation of each part of the signal is calculated. For the region of high deviation, local maxima are extracted. To achieve a strict lossless compression in regions of high standard deviation and a tolerable lossy compression in rest of the signal, two different compression algorithms have been developed. The compression algorithm has been evaluated with the MIT-BIH Arrhythmia Database which revealed that this proposed algorithm can reduce the file size significantly with almost negligible loss of information. By using the reversed logic for reconstruction the data can be reconstructed preserving the significant ECG signal morphology.

Index Terms— creating difference array, local maxima, standard deviation, replacement of critical numbers.

I. INTRODUCTION

An ECG is an important physiological signal for cardiac disease diagnostics. As the sampling rate, sample resolution, observation time and number of leads increase, the amount of ECG data also increases and so the huge storage capacity is required. Specially, when data transmission is required, the amount of the transmission time also increases and it needs more bandwidth for compensation. With all of these limitations, ECG data compression has been one of the most active research areas in biomedical engineering and signal processing.

Techniques for ECG compression can be classified into three categories: (i) transformation methods (e.g., Fourier, Wavelet, KLT, DCT), (ii) parameter extraction method (e.g., long-term prediction, linear prediction) and (iii) direct data compression techniques (AZTEC, CORTES, CM, TP and SAPA, FAN).

Our intension is to achieve maximum reduction of ECG data volume while preserving the significant clinical morphology features upon reconstruction. In the present work, first of all we find the *standard deviation* and *local maxima or peaks* for a particular block of 16 voltage samples of data. ECG data of high deviation region area are compressed without loss and rest of the signal is compressed with negligible loss of clinical information.

In regions of high deviation region, a group of 8 local maxima (i.e. peak) voltage values are taken at a time from the input ECG data file. Sign bit of the corresponding eight voltages values is evaluated.

Now each "Voltage" value is subtracted from its preceding terms to obtain small numbers for getting better compression. These differences are multiplied by a suitable large number to obtain integer number. Those integers are now grouped in both forward and reverse direction maintaining some essential logical criteria. At last these grouped and ungrouped (those not satisfying grouping (criteria) integers along with some essential information (sign bit, position of critical numbers, forward/reverse grouping etc.) are printed in their ASCII character. For all other portion (where the deviation of the signal is low) of the ECG signal, a group of 16 data is taken at a time from input ECG data. Then only 8 voltage values are sorted for later processing. Sign bit of these 8 is generated. These voltages are then multiplied by a suitable large number, grouped only in forward direction and at last printed in their ASCII characters.

In the signal reconstruction domain, two separate de-compressors are used. First for the reconstruction of signal in high SD region and Second for rest of the signal. Signal decompression uses the reverse algorithm.

II. ECG DATA COMPRESSION AND DECOMPRESSION ALGORITHM

Developed algorithm is divided into two sections: i) Data compression, ii) Data reconstruction.

Data compression part is further divided into three sub-sections *(a) standard deviation calculation & Local maxima (Peak) detection, (b) lossless compression for local maxima i.e. peaks using character encoding* (LLCPCE) & (c) *lossy compression for samples other than local maxima(non-peaks) using character encoding*(LCNPCE).

The data reconstruction algorithm is also divided into two sub-parts *(a)) lossless compression for local maxima i.e. peaks* (LLRPCE) *and (b) Lossy reconstruction for non-peaks*(LRNPCE). All these compression reconstruction and feature extraction algorithms are explained sequentially in the rest of the sections. Block schematic of the proposed algorithm is shown in Figure 1.

Manuscript published on 30 April 2013.
***** Correspondence Author (s)

Raj Kumar*, Electronics & Communication Department, NIT Patna, Patna, India.
Sandeep Kumar, Electronics & Communication Department, NIT Patna, Patna,India.
Manish Rai, Electronics & Communication Department, NIT Patna, Patna, India.
Sanket kumar, Electronics & Communication Department, NIT Patna, Patna, India .

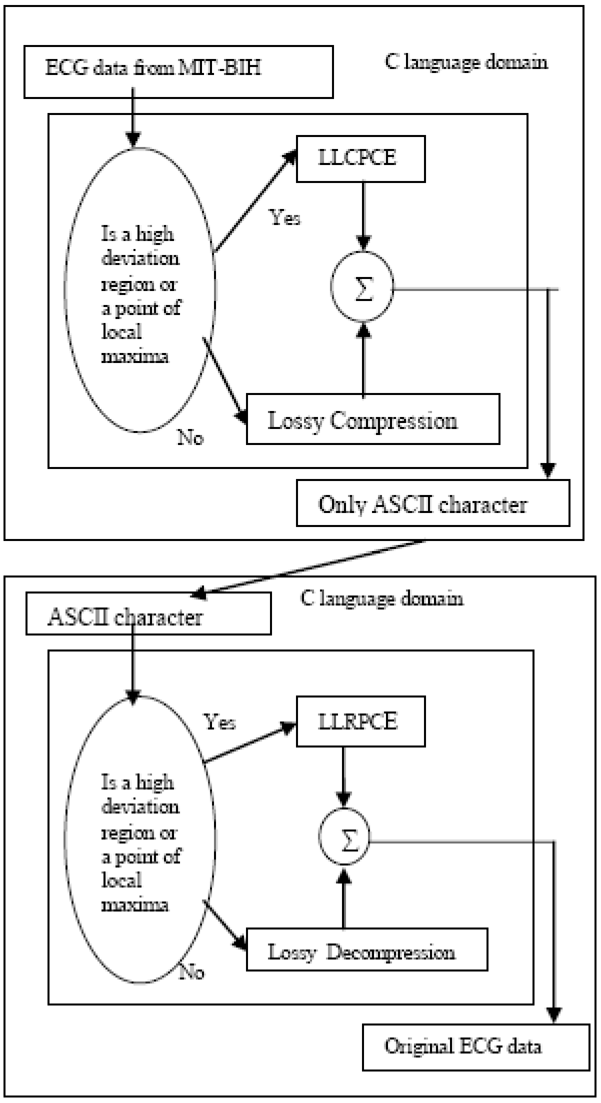


Fig.1. Block Schematic of the proposed algorithm

III. DATA COMPRESSION MECHANISM

Data compression part is divided into three sub-sections (a) standard deviation calculation & Local maxima (Peak) detection, (b) lossless compression for local maxima i.e. peaks using character encoding (LLCPCE) & (c) lossy compression for samples other than local maxima(non-peaks) using character encoding(LCNPCE).

A. Local maxima or Peak detection

i) Standard Deviation Calculation & Local maxima or Peak detection

The algorithm is developed using differentiation technique. From the input ECG data file, at a glance 1000 voltage samples are taken and the 'Mean' is calculated. Now, starting from the first voltage value at a time 16 sample is taken and the Standard Deviation is calculated. If the deviation exceeds a certain threshold of the 'Mean', those voltage samples will be compressed in loss less manner otherwise in lossy approach.. In the ECG waveform different sections of ecg wave have different amplitude as shown in the fig.2 and table no.1. We extract the 8 local maxima or peak voltages for high deviation region from the set of 16 voltage samples. If the no. of peaks is less than 8 bit then we consider next 16 samples and proceed for the set of 32 voltage samples. Those voltage samples will be compressed in loss less manner otherwise in lossy approach. Same process will be continued iteratively for all the data in the input file. For the last time of iteration, there may not present 1000 samples. But the process will continue for those present samples.

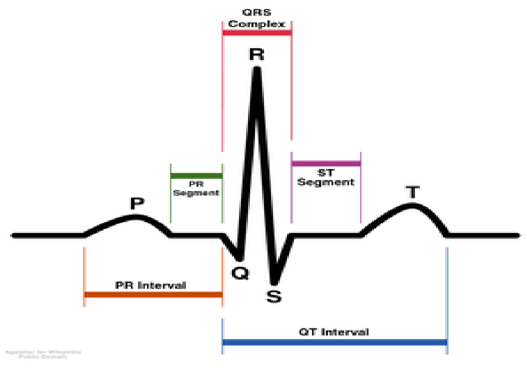


Fig.2 A normal ECG wave

Table 1 Amplitude of different points of ECG signal

point	P	R	Q	T
Amplitude (in mv)	0.25	1.6	25% of R wave	0.1-0.5

ii) Peak position estimation i.e. calculation of indices of peak voltages

Position of peak values for a particular set of ecg data is calculated. For peak values a variable say 'g' is set to 1, and for non-peak values it is set to 0 as sown below.

23	24	21	27	25	26	24	23
G[0] 0	G[1] 1	G[2] 0	G[3] 1	G[4] 0	G[5] 1	G[6] 0	G[7] 0

Hence for the given array value of G will be '01010100' can be obtained whose decimal equivalent is 84. This will be treated as indices of peak voltages.

B. Lossless compression for peak samples using character encoding

To compress peak samples in a lossless manner, following algorithm is developed.

i) Reading ECG Data from Input File

The first objective is to read only 8 'Voltage' values whose amplitude is higher than other voltage samples adjacent to that. By this we find out the point of local maxima. Since the sampling time of the input ECG data is known, the "Time" axis can be easily generated during data reconstruction. Hence concentration is given only to compress the "Voltage" values. To implement this, an array pk[] (say) is declared to hold 8 "Voltage" values at a time. Let in an instant, the content pk[] array is as below.

0.0 23	0.0 25	0.0 22	0.0 24	0.0 30	0.0 34	0.0 35	0.0 31

pk[0]	pk[1]	pk[2]	pk[3]	pk[4]	pk[5]	pk[6]	pk[7]

ii) Creating Difference Array

In order to get better compression a difference array D[] (say) is constructed in such a way that it contains the difference of every elements and its preceding of the pk[] array except the first "Voltage" value which is kept unchanged. The first "Voltage" value is essential at the time of data reconstruction otherwise data reconstruction will not be possible. For the above pk[] array the difference array will be as below

0.023	0.02	-0.03	0.02	0.06	0.04	0.01	-0.04
D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]

iii) Sign beat generation

The sign of the every element of this array D[] is checked and for every positive number a binary zero (0) and for every negative number a binary (1) is taken as sign bit for the corresponding 'Voltage' value. Following this rule, from the elements of the above array a string like "00100001" can be obtained whose decimal equivalent is 33. This will be treated as the sign bit of these corresponding eight 'Voltage' values. Now the sign bit is converted into its corresponding ASCII character and printed in the output file. Now all the numbers in D[] array are multiplied 1000 and also the negative numbers are made positive by multiplying (-1). So all the numbers in are D[] array are now made positive for later processing and stored in another array C[] (say). A sample c[] array is shown below.

23	2	3	2	6	4	1	4
C[0]	C[1]	C[2]	C[3]	C[4]	C[5]	C[6]	C[7]

But the problem arises, if the corresponding ASCII of any decimal equivalent sign bit becomes some special characters like 10 (Line feed), 13 (carriage return), 26 (substitution) or 255 (blank). Those numbers can be printed in character form but at the time of data reconstruction these values (13, 26 and 255) will be considered as 'End of File' by the compiler and the program will get terminated. To avoid this problem, these numbers are substituted by some any other suitable numbers provided an extra bit say (rs) will be sent along with this sign bit which signifies whether the sign bit is changed by the code or not. If the sign bit is changed by the described programming logic, 'rs' will be set to 1 (rs=1) otherwise to 0 (rs=0) and will be printed in character form.

iv) Normalization of C[array]

In the difference array the first number is left unchanged. The problem starts when the ECG data begin with a high voltage (greater than |0.255|). As those numbers after multiplying by 1000 become greater than 255, therefore can't be printed in ASCII character. To overcome this problem, an integer variable say 'ii' is taken and is assigned with the value "First number (C[0]) / 255" and thereafter C[0] is assigned with reminder of this division. Now C[0] can't be larger than 255. There is a probability that 'ii' may contain any of those integers, 10, 13 or 26. If this happens, 'ii' will be replaced by some other suitable number. Finally 'rs' (the sign confirmation bit) is multiplied by 100 and is added with 'ii' and the result is printed in character form.

v) Replacement of critical numbers

Three variables have been taken (say q, r and s initialized with 0) to denote the position of those critical numbers (255, 10, 13, and 26) in C[] array so that at the time of data reconstruction, original value could be bring back. Let in an instant, following numbers are stored in C[] array.

10	5	26	15	26	20	255	13
C[0]	C[1]	C[2]	C[3]	C[4]	C[5]	C[6]	C[7]

After marking the positions of all 'critical' numbers in C[] array, all of those are replaced by some other suitable number.

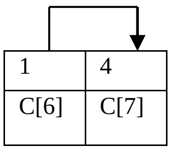
vi) Grouping

It is the most important compression logic throughout the algorithm. Let us take the following for example

Three types of groupings have been done here 1) Forward 2) Reverse and 3) No grouping. Three variables have been taken (k, z and u, initialized with zero (0)) for denoting the positions of these groupings.

a) Forward grouping

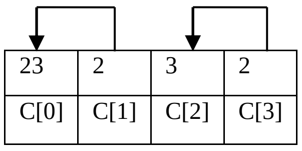
If it is found that any '(C[i]*100)+C[i+1]' is less than 255, then this grouped integer is printed in character form. The variable 'k' is used to denote the forwardly grouped positions.



1	4
C[6]	C[7]

b) Reversed grouping

If forward grouping is not possible then it is checked that whether any '(c[i+1]*100)+c[i]' is less than 255 or not. If so, then this reversed grouped value is printed in character form. 'z' denotes the positions of reversed grouped integers.



23	2	3	2
C[0]	C[1]	C[2]	C[3]

c) No grouping

If both forward and reverse grouping is not possible then, value at those positions are left unchanged as in c[4] and c[5] position. The variable 'u' denotes these positions.

Each set of grouped or not-grouped 'Voltage' values along with the necessary information (sign bit, G, k, z, u etc.) will be printed in character form maintaining the following format. Every character set is separated from each other by an 'ENTER' character.

Sign bit	g	Grouped / Not Grouped values	k	z	u	r	s	q	rs+ii

C. Lossy compression for rest part using character encoding

Rest of the portion of the ECG signal (i.e. except P,R, T wave regions) is compressed in lossy manner. In PTB diagnosis ECG database, the sampling frequency of ECG data is fixed at 1 KHz. The standard clinical bandwidth for the 12-lead clinical ECG is 0.05 – 100Hz . Before passing the original ECG 'Voltage' values into the main LCCE module, the sampling frequency of these regions is reduced to one fourth. This will not violate Nyquist criteria.

a) Reading ECG Data from Input File

For this purpose, at a time 16 'Voltage' values are taken from the input file but only 8 (1st, 3rd, 5th…) of those 16 voltage values are saved for later processing (i.e. reducing sampling freq. to one half). An array e[] is considered to store those 8 'Voltage' values at a time.

b) Sign Bit Generation

Sign bit of these 8 'Voltage' values are generated using the same approach as in case of "LLCPCE".

c) Searching the largest Number and Amplification

In the next step, maximum number from that e[] array is find out and using this maximum number a common amplification factor for those eight "voltage" values is generated in such a way that after multiplication, integer part of each of the "voltage" values will be either less than or equals to nine (9). This amplification factor will also be printed in character form.

d) Grouping

Grouping is done by the same approach as in "LLCPCE" with a little difference. In LLCPCE forward, reverse and no grouping was done but only forward grouping is implemented here.

If any grouped integer becomes 10, 13 or 26 then they are changed to 11, 14 and 27 respectively to avoid the previously described problem. Every set of four grouped 'Voltage' values along with the associated information like Sign-bit, 'rs' and 'Amplification factor' will be printed in character form maintaining the following format.

Sign bit	Aired 'voltage' values (four)	rs	Amp. factor

The above-described LCNPCE algorithm is executed iteratively for compressing all the 'Voltage' values of Low deviating regions.

IV. DATA RECONSTRUCTION ALGORITHM

For reconstruction purpose, three separate decompression algorithms are also developed for decompressing the 'Voltages' decompression of local maxima & for in High deviating area which are lossless and other to decompress the lossy Low deviating area by following reverse logic.

The two sub-parts of data reconstruction algorithm is (a) lossless reconstruction for local maxima i.e. peaks (LLRPCE) and (b) Lossy reconstruction for non-peaks (LRNPCE).

A. Lossless reconstruction for peaks with character encoding (LLRPCE)

a) Reading characters from compressed data file

This module take one set of ASCII value of the compressed ECG data file at a time in an array a[] (say) and then the decompression algorithm is applied on those ASCII values to restore original eight 'Voltage' values. The first ASCII value among those contains the 'Sign bit' of the eight voltage values. Following ASCII values are the grouped / not-grouped integers, grouping positions, position of critical number, and 'rs+ii' respectively. All these are necessary to get back the original eight voltage values. Now from that a[] array all these necessary information are stored separately in different variables. If 'rs' is found to be one (1), signifies that, the 'Sign bit' was changed by the programming logic during LLCPCE and the original is placed back. Form rest of the ASCII values original eight 'Voltage' values will be generated using the following algorithm.

b) Ungrouping

The variable 'k', 'z', and 'u' signify the positions of Forward, Reverse, No grouping respectively. Hence the original eight voltage values are generated using just the reverse logic and stored in an array c[] (say). Let us take an c[] array as below

c) Replacement of Original Numbers

Now 'r','s' and 'q' are taken into account. These variables indicate the position of the critical numbers (10, 13, 26 and 255). Here for example purpose we get r=2. So in c[2]position the original value should be bring back. The same process is applicable for's' and 'q'. Now every time 'ii' is multiplied 255 and added with c[0] also store the result in c[0] position as the reverse was done during LLCPCE, when 'Normalizing' the array.

d) Sign-Bit Regeneration

In this section the Sign-bit will be converted into its corresponding 8-bit binary number and will be saved in an array b[] (say). If any element of the array contains 1 then the corresponding positional element of c[] array will be multiplied by (-1). Here the contained of the elements c[2] and c[7] positions will be multiplied by (-1).

e) De-Amplification

In the next step every number in c[] array will be divided by 1000 and saved in an array d[] (say).

f) Creating Original "Voltage" Values

This d[] array contain the difference between two adjecent voltages. To get the original, each positional contain is added with the previous value. The new values are stored in another array e[] (say). Now by using indices of peak 'g' we can put the peak voltage values at their original position.

g) Generation of Time Axis

A variable ‘x’ (say) is declared and is initialized with zero (0). First time ‘x(=0)’ and e[0] will be printed in the output file. Here the values in e[] array will be treated as ‘Voltage’ values. As we know the sampling time, each time ‘x’ will be incremented by the sampling time and will be printed with the next positional contain of e[] array. This process will continue until e[7] is printed.

The above-described ‘LLRPCE’ algorithm will be executed iteratively in order to decompress local maxima values.

B. Lossy Reconstruction For Rest of the Signal

This module will take seven ASCII values of the compressed ECG data file at a time. Those ASCII numbers will be ungrouped and De Amplified properly as done in “LLRPCE” algorithm. Also the sign of each De amplified voltages are evaluated. At last, after generating the “Time” axis, “Time” and “Voltage” values will be printed in the output file.

V. RESULT

In biomedical data compression, we generally determine the clinical acceptance of reconstructed signal through visual inspection. We may also measure the residual i.e. the difference between the original and reconstructed signal. Such a numerical measure is Percent root-mean-square difference, PRD, given by

$$\text{PRD\%} = \sqrt{\frac{\sum_{i=1}^{N}(Yi - \bar{Y}i)^2}{\sum_{i=1}^{N} Yi^2}} \times 100\%$$

Where y_i represents original signal samples and $\bar{Y}_i$ represents reconstructed signal samples.

The compression ratio (CR), which is defined as below, is also calculated.

$$\text{CR} = \frac{Bo}{Bc}$$

Where B_o is the total number of bits in the original file and B_c is the total number of bits in the compressed file.

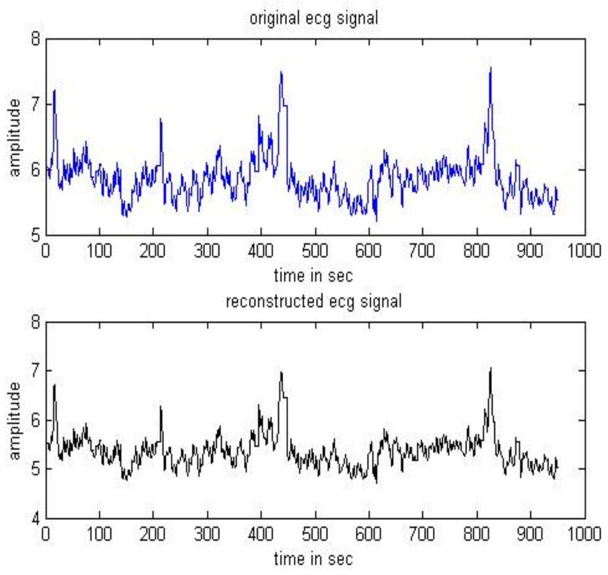


Fig.3: original and reconstructed signal

VI. PERFORMANCE COMPARISON

Table 2. Performance Comparison

Algorithm	**CR**	**PRD%**
m-AZTEC [13]	5.6	25.5
Hilton [14]	8	2.6
Djohan et al. [15]	8	3.9
SPIHT [16]	8	1.8
Shrouf et al. [17]	11.6	5.3s
Perceptual Masks [18]	3.5	1.24
Fira and Goras [19]	12.74	0.61
USZZQ and Huffman coding of DSM [20]	11.06	2.73
S.K Mukhopadhyay et al [7]	7.18	0.023
S.K Mukhopadhyay et al [8]	15.72	7.89
proposed	22.85	7.41

VII. CONCLUSION

We proposed an ECG signal compression method based on ASCII character encoding. Using this very new logic we can obtain CR, which is superior to any of the methods compared above. The performance of the proposed algorithm was compared with some other proposed ECG signal compression methods using two datasets from MIT-BIH arrhythmia database. The method performs well with both normal and abnormal (Myocardial Information) ECG data, and achieves good CR with low distortion and morphological features are well preserved in the reconstructed signal.

REFERENCES

[1] W.J. Tompkins, Editor “Biomedical Digital Signal Processing”.
[2] S. M. S. Jalaleddine, C.. G. Hutchens, R. D. Strattan, and W. A. Coberly, “ECG data compression techniques a unified approach,” IEEE *Trans, Biomed. Eng.*, vol. 37, pp. 329-343, 1990.
[3] A.A.Shrouf, M.A.Zahhad, S.M.Ahmed, A novel compression algorithm for electrocardiogram signals based on the linear prediction of the wavelet coefficients, Digital Signal Process. 13(2003)604–622.
[4] M.L.Hilton Wavelet and wavelet packet compression of electrocardiograms. IEEE Trans Biomed Eng 1997;44(5):394–402.
[5] A.Djohan, T.Q.Nguyen, W.J.Tompkins, ECG compression using discrete symmetric wavelet transform. In: IEEE International conference on Engineering in Medicine and Biology Society, vol. 1.1995. p. 167–8.
[6] Z.Lu, D.Y.Kim, W.A.Pearlman. Wavelet compression of ECG signals by the set partitioning in hierarchical trees (SPIHT) algorithm. IEEE Trans Biomed Eng 1000;47(7):849–56.
[7] S. K. Mukhopadhyay, M. Mitra, S. Mitra, “A lossless ECG data compression technique using ASCII character encoding”, Computers and Electrical Engineering 37 (2011) 486–497.
[8] S.K. Mukhopadhyay, M. Mitra, S. Mitra, “An ECG signal compression technique using ASCII character encoding”, Measurement 45 (2012) 1651–1660.
[9] S.K. Mukhopadhyay, M. Mitra, S. Mitra,”ECG data compression via ASCII character encoding and feature extraction using hilbert transform based approach”,Consumer Electronics Times,vol.2.2013.iss-1,pp 56-67.
[10] John P. Abestein and Willis J. Tompkins, “A new data reduction algorithm For real time ECG analysis”,IEEE Trans Biomed Eng vol.BME 29 1982,pp 43-48
[11] Brian Bradie, “Wavelet Packet Based Compression of Single Lead ECG”,IEEE Trans Biomed Eng ,vol.43 1996,pp 493-501.

[12] Sateh M. S. Jalaleddine,Chris G. Hutchens, Robert D. Strattan and William A Coberly,"ECG data compression techniques-A unified approach",IEEE Trans Biomed Eng, vol 37 1990,pp 329-341.

[13] V. Kumar, S.C. Saxena, V.K. Giri, D. Singh, "Improved modified AZTEC technique for ECG data compression: effect of length of parabolic filter onreconstructed signal", Comput. Electr. Eng. 2005; 31(4–5):334–44.

[14] M.L. Hilton, "Wavelet and wavelet packet compression of electrocardiograms", IEEE Trans Biomed Eng 1997; 44(5):394–402.

[15] A. Djohan, T. Q.Nguyen, W. J. Tompkins, "ECG compression using discrete symmetric wavelet transform", IEEE international conference on engineering in medicine and biology society, Montreal, Que, Canada, vol. 1. p. 167–168.

[16] Z. Lu, D.Y. Kim, W.A.Pearlman, "Wavelet compression of ECG signals by the set partitioning in hierarchical trees (SPIHT) algorithm", IEEE Trans Biomed. Eng. 1000; 47(7), pp-849–856.

[17] A.A. Shrouf, M.A. Zahhad, S.M. Ahmed, "A novel compression algorithm for electrocardiogram signals based on the linear prediction of the wavelet coefficients". Digital Signal Process 2003, 13, pp-604–22.

[18] C.M.D. Rodrigo, M.M. Fabrizia, V.B. Leonardo, "Near-lossless compression of ECG signals using perceptual masks in the DCT domain", CLAIB 2007, IFMBE Proceedings, vol.18, Margarita Island, Venezuela, 2007, pp. 229–231.

[19] C.M. Fira, L. Goras,"An ECG signals compression method and its validation using NNs", IEEE Trans Biomed Eng 55 (4) (2008), pp-1319 1326.

Raj Kumar received his B.Tech in Electronics & Communication Engineering in 2011 from H.N.B. Garhwal University Uttrakhand, India. He has been pursuing his M.Tech in Communication System from NIT Patna, India.

Sandeep Kumar is currently pursuing Master degree from National Institute of Technology Patna. He has his Bachelor degree from Medicap Institute of Technology and Management, Indore, India.

Manish Rai is currently pursuing Master degree from National Institute of Technology Patna. He has his Bachelor degree from IPEC Ghaziabad, UP, India.

Sanket Kumar is currently pursuing Master degree from National Institute of Technology Patna. He has his Bachelor degree from Haldia institute of Technology,Kolkata, India.