Makolo and Lamidi, J Comput Sci Syst Biol 2018, 11:5
DOI: 10.4172/jcsb.1000288

Journal of
Computer Science & Systems Biology

Research Article Open Access

Motif Discovery in DNA Sequences Using an Improved Gibbs (*i*Gibbs) Sampling Algorithm

Makolo AU and Lamidi UA*

Department of Computer Science, University of Ibadan, Nigeria

Abstract

Motifs are repeated patterns of short sequences usually of varying lengths between 6 to 20 bases. Within Deoxyribonucleic Acid (DNA) sequences, these motifs constitute the conserved region of most common signatures for recognizing protein domains that are relevant in it evolution, function and interaction. The Gibbs sampling is a Markov Chain Monte Carlo (MCMC) algorithm which has been applied in the past to discover motifs in DNA sequences. A problem with this technique is the profusion of iterative operations in the sampling process because it progressively chooses new possible motif positions from a continuous randomize sampling in DNA sequences. We applied an Improved Gibbs (*i*Gibbs) sampling algorithm on Breast Cancer (brca) human disease DNA sequences obtained from https://www.ncbi.nlm.nih.gov/nuccore to overcome this unwieldy iteration by altering the processes to obtain a reduced runtime and also achieve an accurate satisfactory motif result. The methodology applied in *i*Gibbs algorithm takes an input of *fasta* or *gbk* DNA file and creates a list of all nucleotides to predict a random sampling starting position. It applies motif length, lesser iterative value and further computes the probability and position ranking scores using Position Weight Matrix (PWM). The algorithm was implemented using *Python, Python(x,y)* and *Biopython.* The *i*Gibbs algorithm was evaluated using varying motif lengths of 12, 18 and 24 on different base lengths of 5,000, 10,000 and 15,000 with different iteration levels. The result showed that the *i*Gibbs returned a better average runtime of 7, 10 and 23 seconds respectively compared to 12, 32 and 60 seconds respectively in the existing Gibbs sampling algorithm found at http://ccmbweb.ccv.brown.edu/gibbs/gibbs.html. The accuracy of the motif result was checked using the hamming distance for finding the contiguous string and minimum edit distance into consensus sequences.

Keywords: Motif discovery; Gibbs sampling algorithm; Breast cancer DNA; Iteration

Introduction

A motif is a consensus pattern common to a set of Deoxyribonucleic Acid (DNA), Ribonucleic Acid (RNA), or protein sequences that share a common biological property, like functioning as binding sites for a particular regulatory protein. Sequence motif discovery is one of the fundamental concerns in bioinformatics that has important applications in locating regulatory sites and drug target identification. It was applied in the study for extraction of structured motifs (several words with well-defined gaps) hence is particularly interesting because of its application to detection of binding sites. Generally, motifs are of varying short length of about 6 to 20 base pairs (bp) [1].

Biological data is interlinked and very complex, thus, managing, accessing and presenting this data in an intelligible form is a critical task. Computer scientists are in the continuous need of creating information systems and tools that will allow biologists to effortlessly make relevant use in simplifying their tasks. The evolution of computer through the enhance speed of parallel programming and execution has drastically increased the development on biological knowledge. There are basically two types of general purpose motif finding algorithms: sample driven approaches identify the locations of the motif occurrences directly while pattern driven approaches take advantage of the assumption that a motif can be specified by a central pattern and use it to reduce the search space [2].

The aim of this study is to improve on the efficiency and performance of Gibbs sampling algorithm for motif discovery in DNA sequences. In order to achieve this, the objectives are: (i) randomizing and selection of motif positions from the DNA sequences using Python, Biopython, and QtDesigner for Graphical User Interface (GUI), (ii) improving the Gibbs Sampling Algorithm using a Position Weight Matrix (PWM) to randomly select possible locations and statistically change those locations to converge at the best possible hidden motif and (iii) the use of lesser iterative loops to achieve an improved runtime and accurate motif result.

The iterations in Gibbs sampler consists of numerous sub-steps and each step having its own transition probability distribution indicating that for an E-dimensional model space each iteration will consist of E subtypes for each parameter [3]. Knowledge can be said to be frightening until we understand it more, this work addresses the relevance of using the *i*Gibbs sampling algorithm whose result is important to solving motif discovery problems by enhancing the knowledge of bioinformatics experts in the areas of design, modeling and prediction from a DNA motif data. Therefore, this study makes motif discovery much comprehensible and less complicated.

Materials and Methods

Materials

In a motif discovery process, the knowledge of utilizing all the different processes as facts to infer the pattern or sequence of the motif in DNA sequence is paramount. These facts are referred to as the central dogma of molecular biology which is mainly focused on

***Corresponding author:** Lamidi UA, Department of Computer Science, University of Ibadan, Nigeria, Tel: +2348068521703; E-mail: uslaad@yahoo.com

Received July 07, 2018; **Accepted** July 31, 2018; **Published** August 02, 2018

Citation: Makolo AU, Lamidi UA (2018) Motif Discovery in DNA Sequences Using an Improved Gibbs (*i*Gibbs) Sampling Algorithm. J Comput Sci Syst Biol 11: 296-305. doi:10.4172/jcsb.1000288

Copyright: © 2018 Makolo AU, et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

how the genetic information contained in the Deoxyribonucleic Acid (DNA) can be processed to provide a meaningful protein sequence [4].

Alignment in the field of bioinformatics is a way of arranging sequences of Deoxyribonucleic Acid (DNA), Ribonucleic Acid (RNA) or protein in order to identify regions of similarity that may be consequences of functional, structural or evolutionary relationship. Very short or very similar sequences can be aligned by hand; however, most interesting problems require the alignment of lengthy, highly variable or extremely numerous sequences that cannot be aligned solely by human effort. Instead, human knowledge is applied in constructing algorithms to produce high quality sequence alignments, and occasionally in adjusting the final results to reflect patterns that are difficult to represent algorithmically (especially in nucleotide sequences) [5].

A Multiple Sequence Alignment (MSA) is used to align multiple sequences at a time; it is an expansion of the pairwise alignment. It allows more than two sequences to be aligned and also the analysis of evolutionary relationship between the many aligned sequences. MSA can be applied as progressive alignment which can be described as a method that manages sequence alignment by passing them through stages continuous alignments. In the selection of the best multiple sequence alignment using the conventional weight matrix, it is assumed that the probability of each base is independent of its neighbouring one [6,7].

Gibbs sampling is a Markov Chain Monte Carlo (MCMC) approach simply known as "Markov Chain", which is a chainlike process. Gibbs Sampling is an iterative procedure that discards one sample after each steps of iteration and replaces it with a new one. It proceeds more slowly and chooses new sample at random increasing the odds that it will converge to the correct solution. The objective will be to identify the best pattern, defined as the most probable motif sequence pattern. A pattern is obtained by locating the alignment that maximizes the ratio of the corresponding pattern probability to background probability. The Markov chain Monte Carlo is a method to sample a given multivariate distribution π^* by constructing a suitable Markov chain with the property that it is limiting; invariant distribution is the target distribution π^* In most problems of interest, the distribution π^* is absolutely continuous and, as a result, the theory of MCMC methods is based on that of Markov chains on continuous state spaces outlined as shown in the handbook of computational statistics [8-10].

In a study of an efficient algorithm for identifying DNA motifs, the problem of combining the voting algorithm and pattern matching algorithm to find exact motifs exists. Therefore, the representation of motif was taken as regular grammar, matrix and string, to compute the hamming distance for finding the contiguous string. The algorithm constructs neighborhood and matching pattern in the sequences to enable selection of a set of possible motif candidate [11].

Motif Containment and Information (MCOIN) algorithm incorporates content as information-based heuristic to automatically determine the most likely motif width as opined by Kilpatrick et al. [12]. The MCOIN makes use of threshold parameter values between 0 and 1 to estimate the true motif width. It also applied the divergence for two discrete probability distributions, to compute Position Weight Matrix (PWM).

Motif extraction mechanism in Makolo et al. [13] termed STGEMS algorithm showed that setting of a threshold p-value position weight matrix, which can be achieved by exhaustive parameter optimization and the use of geometric mean to calculate similarity function (hyper-geometric scoring) was applied with distance metric to extract motifs of *P. falciparum* which is said to have presence of highly repetitive sequences. This further shows a better approach to handling sequences with highly repetitive backgrounds.

In relation to documents topic and word count, the use of Sparse Latent Dirichlet Allocation by Yao et al. [14] to further improve the complexity of Gibbs sampling, by dividing the full conditional probability mass into three parts and employed an approximate sampling scheme to change the document-topic count and word-topic count respectively. Although the relative speedup compared to standard CGS (Collapsed Gibbs Sampling) seems promising. Furthermore, Canini et al. [15] proposed two online inference algorithms: incremental Gibbs sampler and particle filter. In incremental Gibbs sampler, only particular words in the "rejuvenation sequence" are sampled in each of the iteration performed. Thus, the choice of rejuvenation steps determines the runtime of the incremental Gibbs sampler. The particle filter introduced a re-sampling strategy to optimize the Gibbs sampler, however, to implement this algorithm efficiently a special data structure has to be designed and maintained in memory.

The use of dynamic programming sampling strategy in terms of efficiency, convergence and perplexity to significantly improve the efficiency of collapsed Gibbs sampling on large data sets by presenting a parallelization to further improve the efficiency. Collapsed Gibbs Sampling (CGS) is also Markov-chain Monte Carlo Method which is used in Latent Dirichlet Allocation (LDA) for generative integrals in probabilistic generative model. The CGS was used as Efficient Collapsed Gibbs Sampling (ECGS) algorithm in order to speed up the estimating procedure solely to reduce the number of iterations [16].

Oistein [17] made use of the application of Gibbs Sampling in the replacement of counts of single verbs and nouns with the counts of their respective containing clusters, which requires a unique set of clusters as experiment which ran Gibbs sampler 5 times, each time letting the sampler go through 100 iterations before sampling. It finally 'burn-in' in order to let the system reach a state that should not be affected by initialization, then drawing 20 samples from each run with 5 iterations that worked well in prior work. Considering three clusters of samples Si of four elements xi:

$$S_1=\{\{x_1, x_2\}, \{x_3, x_4\}\} \quad (1)$$

$$S_2=\{\{x_1, x_3, x_4\}, \{x_2\}\} \quad (2)$$

$$S_3=\{\{x_1, x_4\}, \{x_2, x_3\}\} \quad (3)$$

Using a linking matrix M=(m_{ij}) for positions *i* and *j*, where:

$m_{i,j}$=1 (if x_i and x_j belong to the same cluster) i.e., a match; and $m_{i,j}$=0 if otherwise,

The matrix M is symmetric by definition and gives the following output:

$$M_2=\begin{bmatrix} .011 \\ .00 \\ .1 \\ . \end{bmatrix}; M_2=\begin{bmatrix} .011 \\ .00 \\ .1 \\ . \end{bmatrix}; M_3=\begin{bmatrix} .001 \\ .10 \\ .0 \\ . \end{bmatrix} \quad (4)$$

$$= M=\frac{1}{3}\sum_i Mi=\frac{1}{3}\begin{bmatrix} .112 \\ .10 \\ .2 \\ . \end{bmatrix} \quad (5)$$

From the equation 5, it results to the averaging of S_1, S_2 and S_3 over the linking matrices. The final clustering S is created by assigning all the pairs of elements x_i and x_j to the same cluster if the corresponding m_{ij} exceeds a threshold.

As described in Ishwaran and Rao [18], the use of filtering (dimension reduction), model averaging and variable selection to approximate subsequent position in the filtering step that will indicate variables to retain probably V_{start} to V_{mid} and variables to filter V_{mid+1} to V_{max} will result in high dimensions yields of an ultra-fast Gibbs sampling procedure with each step of the Gibbs sampler requiring *O(nmax)* operations. Thus, computational effort for the filtering step is linear in both dimensions. The retained variables are then applied to the model implemented by Gibbs sampler, the variable score is calculated with regulation parameters. If applied on DNA sequences using Gibbs sampler the computational speed for discovering motif will be generally rapid as the number of variables at this point will be fraction of the original sequence size.

According to Maksims and Richard [19] Gibbs Sampling was used in the efficient sampling of bipartite problems defined as mapping one set of items to another which are present, with applications ranging from computational biology to information retrieval to computer vision; showed that when using standard Gibbs form to define the probability of a permutation π can be represented as:

$$P(\pi/\theta)=\frac{1}{Z(\theta)}\exp(-\mathrm{E}(\pi/\theta))=\mathrm{Z}(\theta)=\sum_{\pi}\exp(-\mathrm{E}(\pi/\theta)$$

where θ is the set of the model parameters and *E (π, θ)* is the sum of single and/or higher order potentials. The main reason for this is that the path from one probable assignment to another using only pairwise swap is likely to go through regions that have very low probability.

Zeeshan et al. [19] applied another layer of Gibbs sampling to the original algorithm in predictive pattern of patients where sequences are dynamically swapped in and out. Making use of the following techniques:

i. Combinatorial Technique: Searching a set of subsequence $\{m_1, m_2,\ldots m_w\}$ that occurs with similarity and fewer differences within a set of sequences $\{s_1, s_2,\ldots s_n\}$.

ii. Probabilistic: Searching for starting positions $\{p_1, p_2,\ldots p_n\}$ within a given set of sequences $\{s_1, s_2,\ldots s_n\}$ that leads to the best sequence using *A x W* profile matrix (where A is the number of different symbols in the data and W is the length of the motif.

From the techniques the goal is to replace poor matches with potentially better options as to arrive at a cluster of sequences that share conserved motif. Pointing out the challenges that are associated with motif detection in symbolic signals as both skewed distributions related to sparse of abnormal activity which increases number of trivial motifs, the running time of the motif discovery algorithm and motif degeneracy issues encountered in DNA sequences. This indicates that predictive patterns may be preserved and missing in some cases owing to factors like age, gender, clinical history, medications and lifestyle of patients as well as noise that obscures predictive pattern. Numerous computational methods involve estimating the probabilities of alternative discrete choices, often in order to pick the single most probable choice. The basic idea of Gibbs sampling as applicable in this situation simply means operating where there is at least two dimensions and rather than probabilistically picking the next state all at once. Therefore, it will make separate probabilistic choice for each of the *k* dimensions, where each choice depends on the other k^{-1} dimensions.

Owing to the fact that in Gibbs Sampling algorithm there more iteration levels during the sampling process of chain-like operation where each sequence is isolated continuously throughout the operation to obtain results. Higher iteration levels lead to higher runtime of operation. Therefore, this work modifies Gibbs sampling algorithm by using the whole sequence like a python list and perform a random selection for a start position using the length of the supplied motif as a guide. It will then loop through to check for matches and compute the probability ratio base on the distribution by predicting other starting sample positions and making use of lesser iterative levels of sampling process on DNA sequences. The overall rationale is to avoid longer iterative loops in other to obtain a better average runtime and equally arrive at an accurate motif result [20].

Methods

Gibbs Sampling Algorithm has been previously applied to motif discovery. This research adopted Markov Chain Monte Carlo (MCMC) approach to improve motif discovery for an improved runtime result that is obtained through lesser iterations in DNA sequences. The Gibbs Sampling Algorithm will choose the first sequence for sampling. This implies isolation of the first sequence and thus builds probabilistic database on the remaining sequences. In the process it also selects a probabilistic random motif position for each un-chosen sequence putting into consideration the length of the random chosen motif for an appropriate position. Hence, it counts residue occurrences for each position for all un-chosen sequences to build the table that tabulates column-wise the count of the residues as (1 to length of motif) and row-wise for the 4 possible DNA residue identities namely Adenine, Cytosine, Guanine and Thymine (A, C, G and T). Focusing on the un-chosen sequences only and comparing them with the given motif it will lay more emphasis on the probabilistic random motifs aligned with same length, it checks and populates the values representing how many times A, C, G and T appears on each column.

From the un-chosen sequences, further emphasis is applied on the remaining codes excluding the probabilistic random motif, treating these codes all together as one sequence, it searches through for the frequencies of A, C, G and T and tabulate the values in column 0 of the table. Once the table is completely filled, summation of each column is used to get the probabilities of the occurrences of a residue in a given motif position. The probability can be obtained by dividing the value of each cell in a column by the total summation on same column. In a case of a cell value been zero where 0 divide by total summation equals 0 is not applicable, because zero probability results to an error in computation and that should not be allowed to happen, in this case that a cell value is zero does not imply getting zero probability. Therefore, the conventional division sign '/' is not the expected division operation to be used, rather make use of the formula:

$$q_{i,j}=\frac{C_{i,j}+b_j}{N-1+B} \quad (7)$$

Where, $q_{i,j}$=Probability of position *i, j*,

$c_{i,j}$=Value in each cell,

b_j=Pseudo count (Arbitrary number can be 0.5 or 1) for all ACGTs,

N=Total Sum on each column,

B=2 (If b_j=0.5, then B=0.5 × 4 ACGT residues=2).

For the probabilistic given value 0 in the position,

While $A_1=c_{1,j}=0$,

If b_A=0.5,

Then, B=0.5 × 4, i.e., count (ACGT)=2 and N=4

Therefore,

$$P_{1,A} = \frac{C_{1,A} + b_A}{N - 1 + B} = \frac{0 + 0.5}{4 - 1 + 2} = [0.1] \quad (8)$$

Using a value 0 in the formula returns 0.1 instead of zero in a normal division.

The total sum of each probability on each column will result to 1.0 therefore each residue will signify the probability of occurrence it has on each length of the probabilistic random chosen motif from the un-chosen sequences.

Accounting for the background frequency counts for each ACGT on a single column containing the un-chosen codes in the unselected sequences. The mathematical division is also not applicable, hence the formula:

$$q_{0,j} = \frac{c_{0,j} + b_j}{\sum_{i=1}^{j} c_{0,k} + B} \quad (9)$$

Where, $c_{0,j}$ is the value of counts in position 0, b_j is the arbitrary value e.g., 0.5,

$\sum_{i=1}^{j} c_{0,k}$ is the sum of all values in position 0,

B is value of b_j multiply by number of residues ACGTs i.e., 0.5 × 4=2.

After calculating the background frequency counts. From the sequence that was chosen for sampling, the isolated first sequence was chosen for sampling, the un-chosen sequences were used for building the probability table, then calculate weight for each possible motif position in the chosen sequence. With the chosen isolated sequence and the probability table, it applies an iterative process for the sequence.

The algorithm tries to calculate for the probability that the given length of motif is in the 1st, 2nd, 3rd or nth positions. It will multiply the probabilities for each of the positions given a motif position:

$$q_{0,j} = \frac{C_{0,j} + b_j}{\sum_{i=1}^{j} C_{0,k} + B} \quad (10)$$

where, A_1 is the Adenine in 1st position, *pn*, *N*=Probability of Nucleotide in a position, *n* is the position number, while *N* is the Nucleotide Alphabet A, C, G or T. After getting the values for A_1, A_2, A_3 and A_n, it randomly chooses a motif position for the chosen sequence using the weights. This implies trying to normalize them by dividing each position 1 to *n* divided by the sum of the positions, using the formula:

$$A_1 = \frac{P_{1,A} * P_{2,C} * P_{3,G} * P_{4,T} * P_{5,} * \ldots}{P_{0,A} * P_{0,C} * P_{0,G} * P_{0,T} * P_{0,} * \ldots} \quad (11)$$

Where, A_n=A_1, A_2, A_3.

After normalizing, the values obtained as the probability of Adenine starting from first, second, third or nth positions will sum up to 1. It will further reveal the position with the highest probability value.

Finally, it will repeat until convergence; repeating whole process until all sequences has been chosen for sampling using the iteration value provided. It will continue repeating whole process until motif positions for all sequences do not change position anymore. When it obtains some kind of convergence, meaning the position obtained for the first, second, third and nth sequences stops moving, when they stop moving it can probably achieve the best position for the motif in each sequence.

To perform an improved sampling this *i*Gibbs sampling method created a new comprehensive list that contains all the list of nucleotides available in the entire sequence of the file in use. The populated sequence list is applied in satisfying the sampling process which is relevant in predicting a start position of a hidden pattern in a sequence using the length of the motif as a better guide to the sampling operation as shown in Figure 1. Hence the length of the sequence in use is expected to be greater than the length of the motif therefore the length of motif must be lesser than the length of DNA sequence codes in use. Subsequently the use of random positioning to properly select a sequence position that starts from the first index 0 to the length of the first sequence, we also sampled for the next possible position by excluding the same sequence from the computation, this enabled the calculation for every possible position of the motif. A probability was observed to ascertain if the randomly selected start position from amongst all possible selections was a motif or background sequence.

The computed probability for a likely motif position was obtained applying the usual Gibbs Algorithm way of multiplying the probabilities for all available motifs starting point in all positions. The result of the probability obtained is then normalized to a standard value and used in sampling the new position. This modified algorithm applied the use of a burn inward looping method that converges to an appropriate sequence position; it therefore implies that the computation with the higher probability is best suited position for the motif within the sequence. This also indicated the fractions between the actual motif and the background sequences by counting the number of residues occurrence in the selected sequence and the background sequence. Using the brca DNA sequence provided for motif discovery the background count is computed as count of the entire residue occurrence from start point index 0 in the sequence excluding the position of the expected motif. Computed thus: (Increment of Z_N where *N* is nucleotide residues)

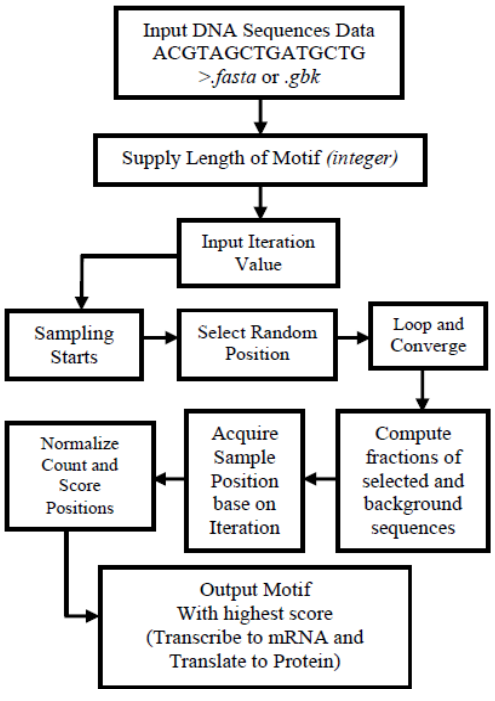

Figure 1: System Design of *i* Gibbs.

Z_N+=1 in Seqz (12)

Where Seq_z is less than starting position [x] or Seq_z greater than the position [x]+len(x).

Hence normalizing the background value is obtained thus: using Z_N divided by $\Sigma(Z_N)$ (summation of all). Whereas for the motif position selected the computation is starting position $[X_N]$ in the sequence with respect to the length of the sequence motif $[X_M]$ where both $[X_N]$ $[X_M]$+=1 increment, the value is also normalized by dividing the count of $[X_N][X_M]$ by the length of the sequence and motif:

$$Q_{XM} = \frac{X_N X_M}{len(\text{Seq}) + \text{len(motif)}} \quad (13)$$

Implementation

The *i*Gibbs system was developed using python programming language which includes the Biopython and Python(x,y) packages. Qt Designer software was used for the interface design and subsequent direct modifications only during the course of implementation as shown in Figure 2. The output of the graphical user interface class was connected into Eclipse Integrated Development Environment (IDE) with the use of an embedded PyDev plug-in feature which enables the coding all features in the python development environment. The input of sequences can be achieved by loading from the file repository directory on the system or from on an online database using the accession number of the sequence file. Python operation always requires the path to a current working directory of any file in use. This system was designed to always return and update the path of a current file in use so as to enable the python smooth operation. DNA Sequences that are pasted into the editor will always need to be saved in *fasta* or *gbk* format to enable same operation. Analysis on the sequence can also be performed to acquire report and summary like total bases, mega bases, total amino acid composition and percentage, GC contents, all nucleotide counts, longest sequence and all codon usage and percentage, about the data content of the sequence file.

A timer was applied using the variables start=time.clock() and stop=time.clock() into the coding section of generate motif operation so as to enable it return the total time elapsed in generating the motif. The timer was computed from the difference of the start and end clocks in seconds. Similarly, a progress bar was used to visually indicate the completion of the generate motif operation from 0 to 100 incrementing at 0.01 level.

Step by Step pseudo code for our *i*Gibbs algorithm is as follows:

Step 1: Load input file,

Step 2: Input Length of motif,

Step 3: Input Iteration Value x,

Step 4: Create a list of DNA Sequence N,

Step 5: Identify k-mer base on motif length,

Step 6: Sampling random position N*i,j*,

Step 7: Loop and converge P*i,j*,

Step 8: Compute selected and background,

Step 9: Acquire motif position in iteration x,

Step 10: Normalize count on PWM and score positions,

Step 11: Output Motif with highest score,

Step 12: End.

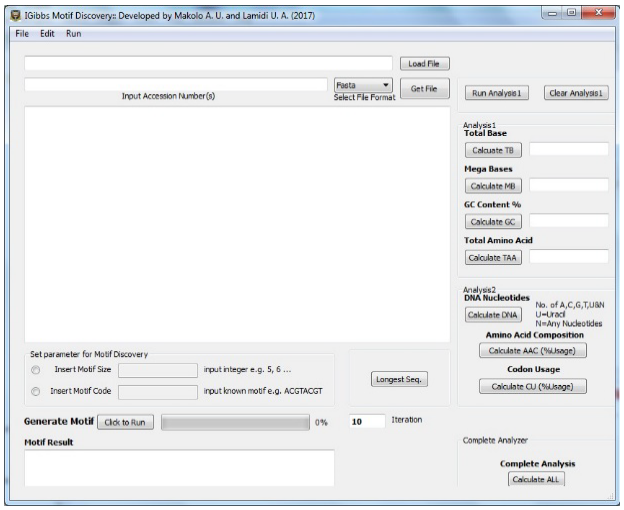

Figure 2: Graphical User Interface design of *i*Gibbs.

Results and Discussion

Once a brca DNA file is properly loaded the implementation is designed such that absolute path of that loaded file automatically displays along with the file content. As for a newly pasted sequence, it can be saved using the menu option on the interface or applying shortcut keys.

As shown in Figure 3 the settings of parameters for, motif length or known motif code and iteration value are all necessary before execution.

In Figure 4 the output of the motif result showing the positions for each motif's starting point within the brca DNA sequence and selecting the best position from the iteration that has best score. It further displays the motif sequences and the time elapsed in generating the motif.

Using the codon table, in Figure 5 the motif result displays the DNA, mRNA and Protein conversion of the generated motif sequence, where the DNA is the most unstable motif sequence from the result, the mRNA changes all Thymine to Uracil and finally the mRNA conversion of AUGACUCUCUAA to protein AUG=Met representing Start, ACU=Thr and CUC=Leu. An asterisk sign * represents a stop codon UAA in the mRNA to protein conversion.

We made use of different parameters namely: total bases ranging from length 5,000, 10,000 to 15,000 brca DNA sequences; motif length of varying sizes 12, 18 and 24, and iteration levels 5 to 15 to achieve repeated results whose outputs each embedded with elapsed runtime we used to compute the average runtime performance of the algorithm as shown in Table 1.

For proper analysis of the result obtained from *i*Gibbs algorithm, we made use of Gibbs motif sampler for brca DNA result online via the website ccmbweb.ccv.brown.edu/cgi-bin/gibbstorun using the same DNA files containing equal number of total bases with same length of motif and recorded the elapsed time for each run as shown in Table 1. The performance of *i*Gibbs algorithm compared to the existing Gibbs sampling algorithm as shown in Table 1 indicated that the total number of bases analyzed were 5,000, 10,000 and 15,000 DNA sequences with several testing operations of the algorithms using the search parameter of motif length 12, 18 and 24 on each of the total bases. During the operation of *i*Gibbs, the iteration level was adjusted using two ranges of

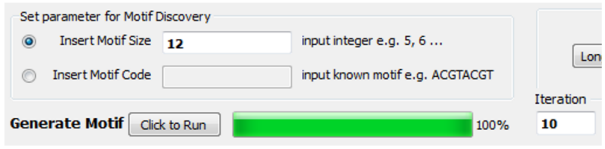

Figure 3: Integer Motif Input and Iteration.

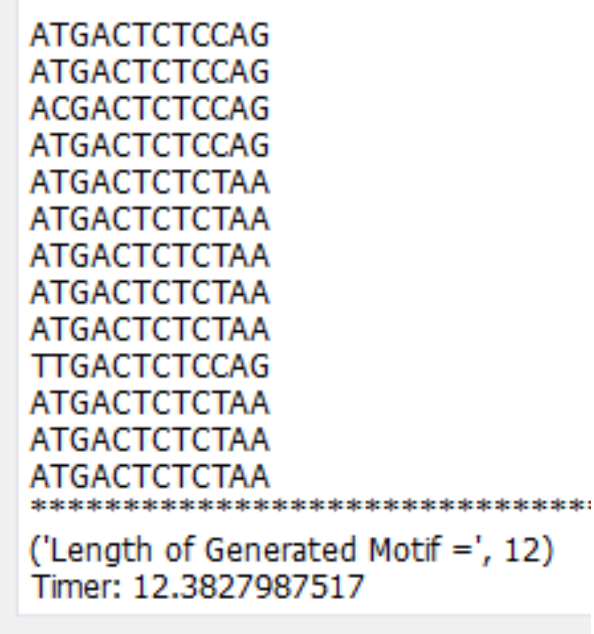

Figure 4: Sample view of Motif Result.

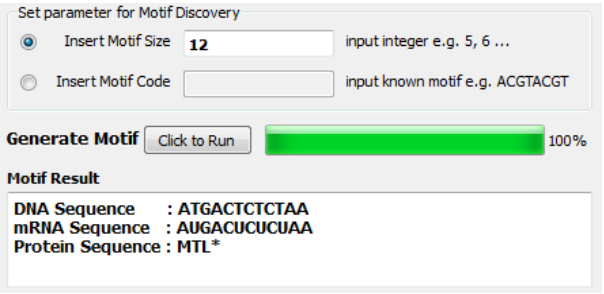

Figure 5: Motif DNA, mRNA and Protein Result.

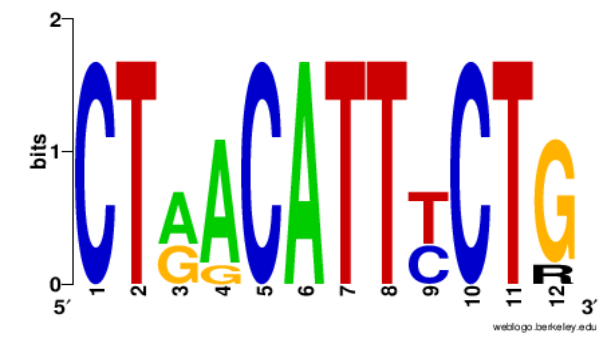

Figure 6: Motif logo 1 using http://weblogo.berkeley.edu/logo.cgi.

5 to 10 and 10 to 15 across all total bases so as to achieve satisfactory run time results. An average runtime was recorded for both the iteration levels used across the three total bases for the *i*Gibbs and existing Gibbs sampling algorithm and we obtained an improved runtime ranging from 3 to 26 seconds in the *i*Gibbs as compared to the existing Gibbs which has an average runtime range of 12 to 60 seconds. The result obtained in Table 1 was further represented on a line graph in Figure for proper comparison and clear representation of the runtime operations between the *i*Gibbs algorithm and the existing Gibbs algorithm.

Beside the runtime, the accuracies for both motif output were observed using the hamming distance as applied in Mostafa and Hazem [1] to calculate mismatches between the set of motif, a record of zero is made for a complete unique column and subsequently a count is made for mismatches found on each column. A total addition of all the value acquired as the mismatch is known as the hamming distance.

From the motif result obtained using *i*Gibbs, we were able to check the accuracy by adopting an approximate match check measured by the use of hamming distance. The Hamming distance is the number of corresponding positions of different bits in the sequence. It simply shows that during the sampling and iteration process the selected motif sequences have minimum edit distance with minimum number of editing operations such as substitution, insertion, deletion needed to transform one or few nucleotides into the other consensus sequences. The *i*Gibbs was able to achieve better time of operation and hamming distance to maintain accuracy of the results.

Further elucidating on the result using the brca1 DNA files as shown in Table 2 that was obtained from https://www.ncbi.nlm.nih.gov/nuccore the following results were obtained using these parameters.

Total base of 5,225 nucleotides, motif size of 12 and iteration level of 5.

**********GENERATING MOTIF**********

(448, 463, 457, 448, 145, 145, 145)1 (326, 341, 335, 326, 717, 331, 331)2

(645, 651, 657, 645, 327, 327, 327)3 (595, 720, 607, 595, 99, 99, 99)4

(10, 19, 19, 10, 715, 715, 715)5

****

('Best Position =', 1) This indicates the selected position from the iteration performed.

['CTAACATTTCTG', 'CTAGCATTTCTG', 'CTAACATTTCTG', 'CTAACATTTCTG',

'CTGACATTCCTG', 'CTGACATTCCTR', 'CTGACATTCCTG']

CTAACATTTCTG

CTAGCATTTCTG

CTAACATTTCTG

CTAACATTTCTG

CTGACATTCCTG

CTGACATTCCTR

CTGACATTCCTG

****

('Length of Generated Motif =', 12) Timer: 3.08549297262 (Time elapsed in seconds for execution)

Total Bases	Length of Motif	*i*Gibbs Algorithm		The Gibbs Motif Sampler (for DNA) Online
		Iterations	Average Runtime (Seconds)	Average Runtime (Seconds)
5,000	12, 18 and 24	5-10	3	12
5,000	12, 18 and 24	10-15	7	
10,000	12, 18 and 24	5-10	9	32
10,000	12, 18 and 24	10-15	10	
15,000	12, 18 and 24	5-10	15	60
15,000	12, 18 and 24	10-15	26	

Table 1: Average runtime of Modified Gibbs Sampling Algorithm compared with Gibbs Motif Sampler Online.

S No	brca1 DNA Sequences
1.	>AY211956.1 Macropus rufus BRCA1 (BRCA1) gene
2.	>AY211955.1 Didelphis virginiana BRCA1 (BRCA1) gene
3.	>AY211954.1 Dendrolagus matschiei BRCA1 (BRCA1) gene
4.	>AY211953.1 Macropus robustus BRCA1 (BRCA1) gene
5.	>KX765637.1 Chodsigoa hypsibia isolate Chypsibi1 BRCA1 (BRCA1) gene
6.	>KX765635.1 Chodsigoa parva isolate Chparva6 BRCA1 (BRCA1) gene
7.	>KX765633.1 Chodsigoa parva isolate Chparva4 BRCA1 (BRCA1) gene

Table 2: 7 BRCA1 DNA sequences obtained from https://www.ncbi.nlm.nih.gov/nuccore.

From the result obtained above the hamming distance has a numerical value of 4 as shown in Table 3 and a web-logo was generated from the consensus sequence as shown in Figure 6.

With the use of same brca1 DNA files as shown in Table 2, the *i*Gibbs operation with the following parameters: Total base of 5,225 nucleotides, motif size of 18 and iteration level of 10 was performed and the following results were obtained.

**********GENERATINGMOTIF**********

(672, 684, 684, 672, 497, 497, 497)1 (175, 254, 184, 175, 712, 712, 712)2

(507, 522, 519, 507, 494, 494, 494)3 (416, 431, 425, 416, 83, 83, 83)4

(3, 12, 12, 3, 315, 315, 315)5 (515, 530, 527, 515, 714, 714, 714)6

(442, 457, 451, 442, 139, 139, 139)7 (333, 348, 342, 333, 316, 316, 316)8

(54, 63, 63, 54, 453, 453, 453)9 (243, 252, 252, 243, 710, 710, 710)10

****

('Best Position =', 4). This indicates the selected position from the iteration performed.

['AGATGGGTATCCAGATAC', 'AGATGGGCATCCAGATGT', 'AGATGGGTATCCAGATAC', 'AGATGGGTATCCAGATAC', 'AGATCTGAATACCGATCC', 'AGATCTGAATACCGATCC', 'AGATCTGAATACCGATCC']

AGATGGGTATCCAGATAC

AGATGGGCATCCAGATGT

AGATGGGTATCCAGATAC

AGATGGGTATCCAGATAC

AGATCTGAATACCGATCC

AGATCTGAATACCGATCC

AGATCTGAATACCGATCC

****

('Length of Generated Motif =', 18) Timer: 4.18014217055 (Time elapsed in seconds for execution)

From the consensus sequence obtained in the result shown above, the hamming distance is a numerical value of 7 as shown in Table 4. A web-logo was generated from the consensus sequence is shown in Figure 7.

Hence, the use of brca1 DNA files as shown in Table 5 which was obtained from https://www.ncbi.nlm.nih.gov/nuccore. The following parameters used on *i*Gibbs: Total base of 10,257 nucleotides, motif size of 12 and iteration level of 5 the following results were obtained:

**********GENERATINGMOTIF**********

(23, 32, 32, 23, 247, 247, 247, 247, 247, 88, 24, 247, 247, 247)1

(2, 11, 11, 2, 314, 314, 314, 314, 314, 155, 298, 314, 314, 314)2

(491, 374, 503, 491, 95, 95, 95, 95, 489, 409, 29, 95, 95, 95)3

(453, 468, 462, 453, 624, 624, 624, 624, 624, 465, 416, 624, 624, 624)4

(2, 647, 11, 2, 314, 314, 314, 314, 314, 155, 298, 314, 314, 314)5

****

('Best Position =', 1) This indicates the selected position from the iteration performed.

['ATTTCTGAGAAG','ATTTCTGGGAAG','ATTTCTGAGAAG', 'ATTTCTGAGAAG', 'TCTCCTGAGAAG', 'TCTCCTGAGAAG', 'TCTCCTGAGAAG', 'TCTCCTGAGAAG', 'TCTCCTGAGAAG', 'ACTCCTGAAAAG', 'ACCCCTGAAAAG', 'TCTCCTGAGAAG', 'TCTCCTGAGAAG', 'TCTCCTGAGAAG']

ATTTCTGAGAAG

ATTTCTGGGAAG

ATTTCTGAGAAG

ATTTCTGAGAAG

TCTCCTGAGAAG

TCTCCTGAGAAG

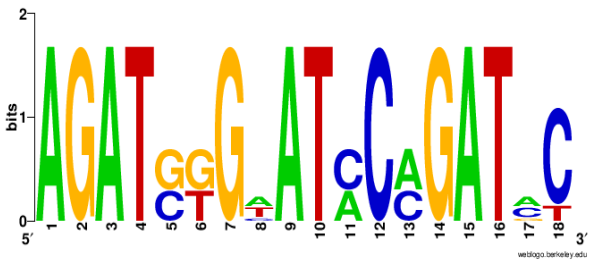

Figure 7: Motif logo 2 using http://weblogo.berkeley.edu/logo.cgi.

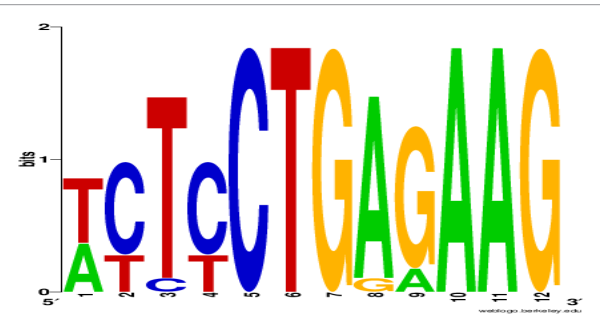

Figure 8: Motif logo 3 using http://weblogo.berkeley.edu/logo.cgi.

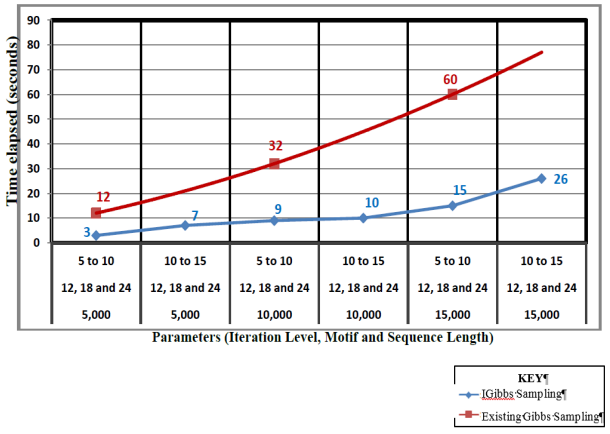

Figure 9: Chart showing runtime of operations.

TCTCCTGAGAAG

TCTCCTGAGAAG

TCTCCTGAGAAG

ACTCCTGAAAAG

ACCCCTGAAAAG

TCTCCTGAGAAG

TCTCCTGAGAAG

TCTCCTGAGAAG

****

('Length of Generated Motif =', 12) Timer: 10.0474499832 (Time elapsed in seconds for execution)

From the consensus sequence obtained in the result shown above, the hamming distance is a numerical value of 6 as shown in Table 6. A web-logo was also generated from the consensus sequence as shown in Figure 8.

Conclusion

This work has been able to contribute to knowledge by applying the use of an improved Gibbs sampling algorithm to successfully extract motifs in DNA sequences. In this work, emphasis was made on improving the operations in the iteration level which was used in the random sampling technique, we have been able to obtain results at better average runtimes range of 3 to 26 seconds using different base lengths of 5,000 to 15,000 brca DNA sequences and varying lengths of 12, 18 and 24 motifs, compared to the existing Gibbs motif sampler algorithm as shown in Figure 9.

The DNA data used in this study are obtained with the use of brcal or ascension number as search keywords. The DNA files are either in *.fasta* or *.gbk* formats and can either be sourced directly from the https://www.ncbi.nlm.nih.gov/nuccore or indirectly from the *i*Gibbs application interface itself using the unique ascension number only. The data was further analyzed to check the general composition of amino acids, GC contents, individual and total nucleotides available in the loaded files that are essential for the motif discovery operation. Hence, about 14 different samples of DNA files were used during the *i*Gibbs operation and each file comprises of varying lengths of nucleotides.

We were also able to maintain accuracy in the motif result obtained with match check by applying the hamming distance calculation. After running *i*Gibbs using motif of length 12 on the 7 DNA files as shown in Table 2 which contains a total base of 5,225 nucleotides, the time elapsed for execution was 3.085 seconds. This operation provided five different starting positions for the consensus pattern in the sequence. From those positions it further made a selection of position (448, 463, 457, 448, 145, 145, 145) 1 as the best consensus pattern with a generated motif logo in Figure 6. The DNA result obtained from the best consensus position was checked using the hamming distance computation between the motif and we obtained numerical value of 4. Using same DNA file in Table 2 with an increase in the motif length to 18 we obtained 10 different possible starting positions and the best position was (416, 431, 425, 416, 83, 83, 83)4 which has the best consensus pattern. The hamming distance was a numerical value of 7 with generated motif logo in Figure 7.

With the use of motif length 12 on Table 5 which has 10,257 nucleotides, we obtained five different possible starting positions and (453, 468, 462, 453, 624, 624, 624, 624, 624, 465, 416, 624, 624, 624) 1 position has the best consensus pattern with a numerical value 6 obtained in the hamming distance and motif logo as shown in Figure 8. Therefore, the application of a *i*Gibbs sampling algorithm using python, biopython and python(x,y) in PyDev eclipse environment was vital in enhancing the performance of the algorithm and also rendered a quality graphical interface with simple usability. We recommend that the analysis can further be expanded to operate on protein sequences.

Declaration of Interest

The authors report no conflicts of interest. The authors alone are responsible for the content and writing of this article.

Acknowledgements

The authors acknowledge the support and encouragement received from Department of Computer Science, University of Ibadan, Nigeria in carrying out this research work.

C	T	A	A	C	A	T	T	T	C	T	G
C	T	A	G	C	A	T	T	T	C	T	G
C	T	A	A	C	A	T	T	T	C	T	G
C	T	A	A	C	A	T	T	T	C	T	G
C	T	G	A	C	A	T	T	C	C	T	G
C	T	G	A	C	A	T	T	C	C	T	R
C	T	G	A	C	A	T	T	C	C	T	G
0	0	1	1	0	0	0	0	1	0	0	1
The hamming distance between the motif=4											

Table 3: Hamming distance of consensus pattern of length 12 from Table 1.

A	G	A	T	G	G	G	T	A	T	C	C	A	G	A	T	A	C
A	G	A	T	G	G	G	C	A	T	C	C	A	G	A	T	G	T
A	G	A	T	G	G	G	T	A	T	C	C	A	G	A	T	A	C
A	G	A	T	G	G	G	T	A	T	C	C	A	G	A	T	A	C
A	G	A	T	C	T	G	A	A	T	A	C	C	G	A	T	C	C
A	G	A	T	C	T	G	A	A	T	A	C	C	G	A	T	C	C
A	G	A	T	C	T	G	A	A	T	A	C	C	G	A	T	C	C
0	0	0	0	1	1	0	1	0	0	1	0	1	0	0	0	1	1
The hamming distance between the motif=7																	

Table 4: Hamming distance of consensus pattern of length 18 from Table 1.

S No	BRCA1 Sequences
1.	>AY211956.1 Macropus rufus BRCA1 (BRCA1) gene
2.	>AY211955.1 Didelphis virginiana BRCA1 (BRCA1) gene
3.	>AY211954.1 Dendrolagus matschiei BRCA1 (BRCA1) gene
4.	>AY211953.1 Macropus robustus BRCA1 (BRCA1) gene
5.	>KX765637.1 Chodsigoa hypsibia isolate Chypsibi1 BRCA1 (BRCA1) gene
6.	>KX765635.1 Chodsigoa parva isolate Chparva6 BRCA1 (BRCA1) gene
7.	>KX765633.1 Chodsigoa parva isolate Chparva4 BRCA1 (BRCA1) gene
8.	>KX765631.1 Chodsigoa parva isolate Chparva1 BRCA1 (BRCA1) gene
9.	>KX765630.1 Chodsigoa hypsibia isolate Chypsibi3 BRCA1 (BRCA1) gene
10.	>DQ630209.1 Sorex minutus isolate 1 BRCA1 (BRCA1) gene
11.	>KF758458.1 Sminthopsis crassicaudata BRCA1 (BRCA1) gene
12.	>KX765618.1 Chodsigoa furva isolate Cfurva2 BRCA1 (BRCA1) gene
13.	>KX765621.1 Chodsigoa hypsibia isolate Chypsibi5 BRCA1 (BRCA1) gene
14.	>KX765614.1 Chodsigoa smithii isolate Chsmithii7 BRCA1 (BRCA1) gene

Table 5: 14 BRCA1 DNA sequences obtained from https://www.ncbi.nlm.nih.gov/nuccore.

A	T	T	T	C	T	G	A	G	A	A	G
A	T	T	T	C	T	G	G	G	A	A	G
A	T	T	T	C	T	G	A	G	A	A	G
A	T	T	T	C	T	G	A	G	A	A	G
T	C	T	C	C	T	G	A	G	A	A	G
T	C	T	C	C	T	G	A	G	A	A	G
T	C	T	C	C	T	G	A	G	A	A	G
T	C	T	C	C	T	G	A	G	A	A	G
T	C	T	C	C	T	G	A	G	A	A	G
A	C	T	C	C	T	G	A	A	A	A	G
A	C	C	C	C	T	G	A	A	A	A	G
T	C	T	C	C	T	G	A	G	A	A	G
T	C	T	C	C	T	G	A	G	A	A	G
T	C	T	C	C	T	G	A	G	A	A	G
1	1	1	1	0	0	0	1	1	0	0	0
The hamming distance between the motif=6											

Table 6: Hamming distance of consensus pattern of length 12 from Table 5.

References

1. Mostafa MA, Hazem MB (2013) An Efficient Algorithm to Identify DNA Motifs. Math Comput Sci 7: 387-399.
2. Sing-Hoi S, Xiaoyan Z (2007) Improved Pattern-driven Algorithms for Motif Finding in DNA Sequences. Springer-Verlarg Berlin. Heidelberg.
3. Jonathan MK, George YS, Dirk PK (2007) The Generalized Gibbs Sampler and the Neighborhood Sampler.
4. Warf MB, Berglund JA (2010) Role of RNA structure in regulating pre-RNA splicing. Trends in Biochemical Sciences 35: 169-178.
5. Polyanovsky VO, Rothberg MA, Tumanyan VG (2011) Comparative analysis of the quality of a global algorithm and a local algorithm for alignment of two sequences. Algorithms Mol Biology 6: 25.
6. Agrawal A, Huang X (2011) Pairwise Statistical Significance of Local Sequence Alignment Using Sequence Specific and Position Specific Substitution Matrices. IEEE/ACM Transactions on Computational Bio and Bioinformatics 8: 194-205.
7. Anthony JFG, Susan RW, Sean BC, John D (2011) Introduction to Genetic Analysis. WH Freeman Publisher.
8. Muhannad AA, Nur'aini AR, Rosni A, Awsan AH, Atheer AA (2015) Investigation Study: An Intensive Analysis for MSA Leading Methods. Journal of Theoretical and Applied Information Technology 75: 1-12.
9. Jian-Jun S, Kian YY, Weng KC (2012) An Improved Scoring Matrix for Multiple Sequence Alignment Mathematical Problems in Engineering 2012: 1-9.
10. Liu D, Xiong X, Das Gupta B, Zhang H (2006) Motif discoveries in unaligned molecular sequences using self-organizing neural network. IEEE Transactions on Neural Network 17: 919-928.
11. James EG, Yuichi M, Wolfgang KH (2012) Handbook of Computational Statistics. Springer, Heidelberg, Dordrecht, London, New York, p: 76.
12. Kilpatrick AM, Bruce W, Stuart A (2013) MCOIN: A novel heuristic for determining transcription factor binding site motif width. Algorithms Mol Biol 8: 16.
13. Makolo A, Ezekiel A, Osofisan A (2012) Comparative Analysis of Similarity Check Mechanism for Motif Extraction. African Journal of Computer & ICT 5: 53-58.
14. Yao DM, McCallum A (2009) Efficient methods for topic model inference on streaming document collections. In SIGKDD, pp: 937-946.
15. Canini KR, Shi L, Griffiths TL (2009) Online inference of topics with latent Dirichlet allocation. In AISTATS, p: 5.
16. Han X, Thomas S (2010) Efficient Collapsed Gibbs Sampling for Latent Dirichlet Allocation. JMLR: Workshop and Conference Proceedings 13: 63-78.
17. Oistein EA (2011) Grammatical Error Prediction. Technical Report. University of Cambridge Computer Laboratory.
18. Ishwaran H, Rao JS (2010) Generalized Ridge Regression: Geometry and Computational Solutions when p is larger than n. America Math Society 622: 81-93.
19. Maksims NV, Richard SZ (2012) Efficient Sampling for Bipartite Matching Problems. University of Toronto.
20. Syed Z, Stultz C, Kellis M, Indyk P, Guttag J (2010) Motif Discovery in Physiological Datasets: A Methodology for Inferring Predictive Elements. ACM Trans Knowledge Discovery Data 4: 2.