Finding Exact and Solo LTR-Retrotransposons in Biological Sequences Using SVM

Torabi Dashti, Hesam; Masoudi-Nejad, Ali*+; Zare, Fatemeh

Laboratory of Systems Biology and Bioinformatics (LBB), Institute of Biochemistry and Biophysics and Center of Excellence in Biomathematics, University of Tehran, Tehran, I.R. IRAN

ABSTRACT: *Finding repetitive subsequences in genome is a challengeable problem in bioinformatics research area. A lot of approaches have been proposed to solve the problem, which could be divided to library base and de novo methods. The library base methods use predetermined repetitive genome's subsequences, where library-less methods attempt to discover repetitive subsequences by analytical approaches. In this article we propose novel de novo methodology which stands on theory of pattern recognition's science. Our methodology by using Support Vector Machine (SVM) classification and clustering methods could extract exact and Solo LTR-retrotransposons. This methodology issued to show complexity efficiency and applicability of the pattern recognition theories in bioinformatics and biomathematics research areas. We demonstrate applicability of our methodology by comparing its results with other well-known de novo method. Both applications return classes of discovered repetitive subsequences, were their results when had applied on show more that 90 percents similarities.*

KEYWORDS: *LTR, Support vector machine, DNA repeact, Repetitive sequence.*

INTRODUCTION

Transposable Elements (TEs) are genetic elements that can transpose from one location to another within the genome. Retrotransposons are a major component of eukaryotic genomes [2]. For example, at least 40 percent of the human genome is composed of retrotransposons; this emerges to criticality of having efficient and reliable methods for discovering these components. These elements classified into two categories; the first is retroelements, which are transposed through the reverse transcription of an RNA template, and other one is DNA transposons, which are transposed through a classical DNA "cut-and-paste" transposition model. These categories divide to some classes where each class has specified properties. One of the DNA transposons classes is LTR Retrotransposons. LTR-Retrotransposons have a unique structural feature. They compound of two direct Long Terminal Repeats (LTRs) that range from -100 bp to over 5 kb in size. The LTRs flank the internal coding region. Most of the proposed methods for annotating TEs in whole genome are based on homology searching against predetermined TEs that stored in databases. One of these methods is LTR-STRUC [3] which is common application. LTR-STRUC searches for finding significant matches between genes against the databases entries. This methods disability for recognizing new TE and also the concept of finding match which backs to theory of

***** *To whom correspondence should be addressed.*
+ *E-mail: amasoudin@ibb.ut.ac.ir*
1021-9986/12/2/111 *6/$/2.60*

sequence alignment and its associated time complexity, attracted interests to de novo methods. Some de novo methods have been introduced for automating repeat elements in a genome [4]. RepeatScout [5], PILER [6] and a combinatorial method [7] are well-known methods which attempted to identify repeat elements based on their copy numbers in a genome, thus facilitating identification of general repeat elements had done. Many TEs indeed appear high copies in the host genome because of their transposition activity. But some TEs families have low copy numbers in some genomes. For example many low copy repeats in mammalian genomes are induced by segmental duplications. *Pevzner* [1] proposed an algorithm which solved the problem as kind of linguistic problems. Solution divided the problem into three separate small problems. The first problem is discovering exact matched subsequences; the second focused to clustering determined subsequences and finally, the third problem is discovering similar -Not exact matched- subsequences. We propose a methodology that uses three approaches regarding to [1] division. Our methodology composed of suffix tree data structure and one of the powerful and practical machine learning approaches in order to tackle the small problems to solve the biggest one. We apply two well known methods from pattern recognition area based on Support Vector Machines (SVM); Support Vector Clustering (SVC) and Support Vector Machines Classifiers (CVMC) which introduced by *Vapnik* [8, 9]. The methods introduced in order to increase accuracy and moreover reducing running time complexity for inferencing information from specified patterns.

EXPERIMENTAL SECTION

As mentioned [1] approach consist of three steps and we develop our methodology based on the steps' definition. Following we describe our solution for each step respectively. Before illustrating the proposed solutions, following, short overviews on SVC and SVMC are presented.

Preliminaries, A short overview on SVC and SVMC

Support Vector Machines (SVM) as a class of statistical machine learning methods, try to reduce the structural risk principle in opposite of empirical risk minimization. This feature of SVM equips it with strong potential to apply on wide range of pattern analysis problems [10]. This class proposed by *Vapnik et al.* [9] for two class classification problem. Thereafter the SVM had extended to solve multi-class classification problems. The basic of SVMC rely on preprocessing the training data points and tries to design a hyperplane which could separate training data points correctly. For any new given data point, SVMC measure distance of the data point with determined hyperplane to find its associated cluster. The training data points are labeled entries where the label indicates the entry cluster. Measuring process was performed by a distance function, called Kernel function. Whereas the Kernel function would measure objects' distance, there are a lot of different Kernel functions according to objects' type and kind of their distribution. The accuracy of SVMC is strongly related to the Kernel function and surely, like other types of learning methods, to the training data points which should represent their clusters clearly. In any kind of problems, there are two types of data points; the first which would be separated linearly and those which are non-linearly separable. There is a fact for non-linear separable data points, that by increasing data points' dimension, they could be separated via a linear function. SVMC uses this fact and apply a map function to inject data points to a higher dimensional subspace from the Hulbert space, where suits Kernel function supposed to separate data points linearly. The map function φ: $R_d \rightarrow H$ should satisfy some constrains which are indicated in [11]. SVM process graphically depicted in Fig. 1.

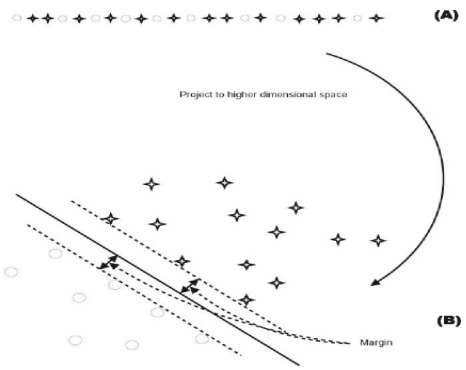

Fig. 1: Two-class objects are indicated in picture, where in section (A) they are not linearly separable but via projection to high dimensional space continues black line could separate them linearly with maximum Margin area.

Via Parzen window *Vapnik* like [12] estimated probability density, and proposed a clustering method [8] on foundation of SVM. SVC is unsupervised learning method which uses the map function and transport data points to higher dimensional subspace where SVC can search for fitted sphere that encloses data points. Analysis of distances of encircled data points yield to data points' different clusters. SVC also uses Kernel function to measure the distances. Fig. 2 shows four steps of sample clustering process [8].

Step one, Finding exact matched subsequences

Whereas, finally the method would extract similar subsequences, in this section a trick for finding Roughly Exact Matched (REM) subsequences could be helpful. Two sequences are roughly matched when some substitutions had happened on sequences. Substituted subsequences started at equal positions index on sequences and also their lengths are smaller than Lengths of Allowed Substitutions (LAS). Fig. 3 indicates a sample of REM sequences.

Therefore, in this scenario we discover exact matched subsequence and thereafter adjunct every two exact matched subsequences where separated by a gap with length smaller than the LAS. The LAS indicates accuracy of the process in reverse relation where large LAS has low accuracy and vice versa. Following, discovering exact matched subsequences via suffix tree is illustrated in brief and at the end of this section we describe how we adjunct the determined subsequences together to reach long REM strings.

Constructing such a tree for the string S takes linear time and space of the length of S [13- 16]. Fig. 5 shows T(S) where S="babab" on $\Sigma = \{a, b\}$.

Finding Exact Matched strings

Biological rules of mobile sequences show that some of them enclose specified sequences. For example LTR-Retrotransposons' sequences enclose coding regions. Therefore to mobile sequence discovery, exact matched sequences which are far away from each other- According to biological rules- must be determined.

As mentioned we used suffix tree data structure in this step. *Gusfield* [17] proposed a linear time algorithm for suffix tree construction on *Ukkonen* [15] algorithm. Subsequently *Lyngso* [18] improved the *Gusfield*'s algorithm for pattern finding problem and demonstrated

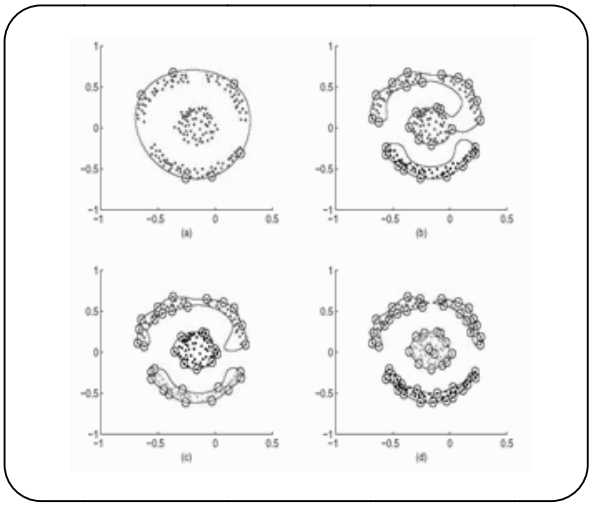

Fig. 2: Clustering of a data set containing 183 data points. Different clusters are represented by different colors. (8).

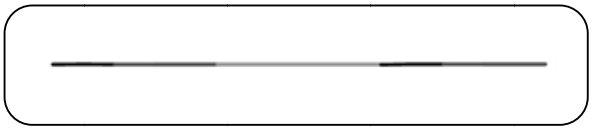

Fig. 3: Sample of REM sequences, where non-matched string, red lines, separates two exact matched parts, black and blue.

that suffix tree data structure suits for discovering repeated strings where the strings bind limited gap. Lyngso's bounded gap is what is needed for distance of exact matched sequences. Here, we apply Lyngso method for exact matched sequence discovery.

This section uses [18] notations. S denotes a given string on finite set Σ as alphabet where the length of S is n. S[i] {for i=0...n} indicates S's i'th character. We use S[i...j] to denote a substring of S that shows S[i]S[i+1] ... S[j] characters.

Definition 1, Repeated string

(i, j, ||α||) indicates a repeated area on *S* with length α where $1 < i < j < ||S|| - ||\alpha||$ and S[i, i+ ||α||] = S[j, j+ ||α||]. This means that if the sequence α copied at positions i and j on S then S has a pair of length ||α||. Fig. 4 shows a pair and its enclosed gap between repeated regions.

Definition 2, Suffix Tree

Suffix tree for sequence S is a tree T(S) whose edges is labeled with strings, such that each suffix of S

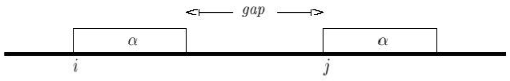

Fig. 4: The subsequence α repeated at positions i and j.

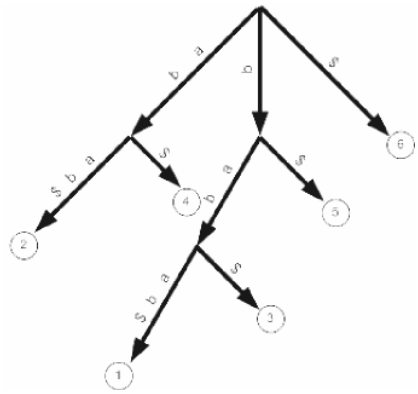

Fig. 5: The suffix tree of S="babab".

corresponds to exactly one path from the tree's root to a leaf (by concatenating edges' labels).

Thereafter, when the sequence's suffix tree is constructed, Lyngso's method make an AVL-Tree for the leafs. Moreover, by performing a common tree traversal algorithm like post order traversal, Lyngso extracted repeated subsequences which bind a gap.

Therefore by using Lyngso algorithm our specific exact matched sequences would be discovered.

Adjunction of determined exact matched subsequences

As mentioned to find a REM sequence we should find exact matched subsequences which bind a gap. Assume previous algorithm stores exact matched sequences as a link list A such that each node {'a'— a $\subseteq$ A} denotes a pair (i_a, j_a, $||\alpha_a||$).

Here, we sort the link list's nodes on their starting point, ia, and traverse it. During the traversing process, each two continuant nodes 'b'=(i_b, j_b, $||\alpha_b||$) and 'a'=(i_a, j_a, $||\alpha_a||$) which satisfy following constrains would be adjunct together.

$i_b - i_a - ||\alpha_a|| = \sigma$;

$j_b - j_a - ||\alpha_a|| = \sigma$;

Where σ is length of bounded gap.

Traversing process tries to find exact matches which can satisfy the constraint's conditions. Attention that there is not any limitation on number of continued substring. for example if there were three nodes a, b, c where $i_a < i_b < i_c$ and a and b satisfy the constrains and also b and c do; at first, the algorithm merges a and band replace their nodes in A with a new node X=(i_a, j_b, $||\alpha_a||+||\alpha_b||+||\sigma||$) and moreover X and c will be merged to reach a bigger REM d = (i_a, j_c, $||\alpha_a||+||\alpha_b||+||\alpha_c||+2\times||\sigma||$).Finally the longer REM d will be placed instead of X and c, and traversing process kept on.

This section's algorithm indicated following.

Step two, Clustering determined subsequences:

Sequence alignment method is basic of common sequence clustering approaches which suits for measuring sequences' distances [19- 21]. In this section, we also use sequence alignment method, but just as a part of SVC Kernel-Table of sequences alignments' score was prepared via preprocessing computation, and feed to SVC-. Our SVC Kernel K(X, Y) is instantiation of Radial Basis Function (RBF) based on sequences' alignment score. K(X,Y) determines measure of X and Y by Eq. (1).

$$K(X, Y) = e^{\lambda \times AScore(X, Y)} \quad (1)$$

The A is real positive number to scale the Kernel value for numerical stability and A score function returns the sequences' alignment score.

Whereas regular local or global sequence alignment methods are not eligible for measuring sequences distance for clustering process, we modified local sequence alignment method. Regular local sequence alignment method returns best score of aligned subsequences, where instead of best score our modified method returns sum of scores of local aligned subsequences.

The foundation of support vector clustering proposed by [8] was defined for distributed data points on Gaussian distribution formula. Whereas [*22*] demonstrated, estimating Gaussian distribution for distribution of repeated subsequences on genome is not so bad estimation, SVC can professionally determine smooth clusters accurately.

Step three, Discovering similar subsequences

In this section, we use the results of previous sections to discover Solo LTR subsequences. Here, minimum

length of exact matched subsequences, L, is needed. Where to discover similar subsequences, whole of genome's L-mer's are examined to find their associated clusters. The clusters are those which determined in step two.

Applying profile hidden markov model [5] or alignment method [23] are samples of introduced methods for discovering similar subsequences which need a lot of time to process. Support Vector Machines Classifiers (SVMC) is practical classifier which after preprocessing training data set could classify new given data point in linear time. The classification accuracy strongly related to the SVMC's Kernel function and training data set's entries.

To discover similar subsequences, we apply SVMC on section two's clusters as training data points and thereafter all of genome's Lmer's which have stored inside of constructed suffix tree would be classified to the clusters.

CONCLUSIONS

This paper is arranged to introduce computationally efficient, accurate, and practical method Support Vector Clustering (SVC) to the field of genome sequence analysis, specially discovering repetitive subsequences. Our method evaluated on simple data where its accuracy on the data base is notable. In this case, the applied Kernel functions are simple as possible just to show the ability of the method to differentiate classes, and it is clear that determining the functions based on distribution of given data base's entries should improve the accuracy.

In addition, SVC is strongly independent to predetermined repetitive subsequences (libraries) and also numbers of sequences' repetition, therefore SVC is eligible to discover new repetitive subsequences with low number of repetitions. Whereas SVC is unsupervised clustering approach and by using it, monitoring number of classes and level of classes' entries' dependency are so handleable parameters, we believe there is great potential for extending the method for other sequence analysis problems in biological systems. Designing suitable Kernel functions or combining the SVC with other analyzers could yield to perfect method to attack to any kind of sequence analysis problems.

Acknowledgement

Part of this work has been supported by Iran National Science Foundation (http://www.insf.org).

Received : Apr. 18, 2010 ; Accepted : July 22, 2011

REFERENCES

[1] Pevzner P.A., Tang H., Tesler G., De Novo Repeat Classification and Fragment Assembly, *Genome Res.*, **14**, p. 1786 (2004).

[2] Kumar A., Hirochika H., Application of Retrotransposons as Genetic Tools in Plant Biology, *Trends in Plant Sciences,* **6**, p. 127 (2001).

[3] McCarthy E., McDonald J., LTR-STRUC: a Novel Search and Identification Program for LTR Retrotransposons, *Bioinformatics,* 12; **19**(3), p. 362 (2003).

[4] Bao Z., Eddy S., Automated de Novo Identification of Repeat Sequence Families in Sequenced Genomes, *Genome Res*., **8**, p. 1269 (2002).

[5] Price A.L., Jones N.C., Pevzner P.A., De Novo Identification of Repeat families in Large Genomes, *Bioinformatics.*, **21**, p. i351 (2005).

[6] Edgar R.C., Myers E.W., PILER: Identification and Classification of Genomic Repeats, *Bioinformatics,* 21 Suppl 1, p. i152 (2005).

[7] Rho M., Choi J.-H., Kim S., De Novo Identification of LTR Retrotransposons in Eukaryotic Genomes, *BMC Genomics,* 3;8:90 (2007).

[8] Ben-Hur A., Horn D., Siegelmann H.T., Vapnik V., Support Vector Clustering, *Journal of Machine Learning Research,* **1**, p. 125 (2001).

[9] VapnikV., "The Nature of Statistical Learning Theory", Springer-Verlag Press (1995).

[10] Cristianini N., Shawe-Taylor J., "An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods", Cambridge University Press (2000).

[11] Duda R.O., Hart P.E., Stork D.G., "Pattern Classification", 2nd Ed., Elsevier Academic Press; (2003).

[12] Roberts S.J., Non-Parametric Unsupervised Cluster Analysis, *Pattern Recognition*, p. 30, 261272 (1997).

[13] Farach M., Optimal Suffix Tree Construction with Large Alphabets, "Annual Symposium on Foundations of Computer Science", pp. 137-143 (1997).

[14] McCreight E.M., A Space-Economical Suffix Tree Construction Algorithm, *Journal of the ACM,* **23**(2), p. 262 (1976).

[15] Ukkonen E., On-Line Construction of Suffix Trees, *Algorithmica,* **14**, p. 249-260 (1995).

[16] Weiner P., Linear Pattern Matching Algorithms, In: "Proceedings of the 14th Symposium on Switching and Automata Theory", pp. 1-11 (1973).

[17] Gusfield D., "Algorithms on String, Trees, and Sequences", Cambridge University Press (1997).

[18] Gusfield D., "Algorithms on String, Trees, and Sequences, Computer Science and Computational Biology", Cambridge University Press (2005).

[19] Agarwal P., States D., The Repeat Pattern Toolkit (RPT):Analyzing the Structure and Evolution of the C. Elegans Genome, *Proc.Int. Conf. Intel. Syst. Mol. Biol.,* **2**, p. 1 (1994)

[20] Kurtz S., Ohlebusch F., Schleiermacher C., Stoye J., Giegerich R., Computation and Visualization of Degenerate Repeats in Complete Genomes, *Proc. Int. Conf. Intel. Syst. Mol. Biol.*, **8**, 228238 (2000).

[21] Burke J., Davison D., Hide W., d2-Cluster: A Validated Method for Clustering EST and Full-Length cDNA Sequences, *Genome Res.,* **9**, p. 1135 (1999).

[22] Malde K., Schneeberger K., Coward E., Jonassen I., RBR: Library-Less Repeat Detection for ESTs, *Bioinformatics*, **22**(18), p. 2232 (2006).

[23] Huang X., Wang J., Aluru S., Yang S.P., Hillier L., PCAP: A WholeGenome Assembly Program, *Genome Res.*, **13**, p. 2164 (2003).