# Boosting the Detection of Transposable Elements Using Machine Learning

Tiago Loureiro[1], Rui Camacho[2], Jorge Vieira[3], and Nuno A. Fonseca[4]

[1] DEI & Faculdade de Engenharia, Universidade do Porto, Portugal
tiagodloureiro@gmail.com
[2] DEI & Faculdade de Engenharia & LIAAD-INESCTEC, Universidade do Porto, Portugal
rcamacho@fe.up.pt
[3] IBMC - Instituto de Biologia Molecular e Celular & Universidade do Porto, Portugal
jbvieira@ibmc.up.pt
[4] EMBL Outstation, European Bioinformatics Institute (EBI), Hinxton,
Cambridge CB10 ISD, UK
CRACS-INESCTEC, Portugal
nf@ebi.ac.uk

**Abstract.** Transposable Elements (TE) are sequences of DNA that move and transpose within a genome. TEs, as mutation agents, are quite important for their role in both genome alteration diseases and on species evolution. Several tools have been developed to discover and annotate TEs but no single one achieves good results on all different types of TEs. In this paper we evaluate the performance of several TEs detection and annotation tools and investigate if Machine Learning techniques can be used to improve their overall detection accuracy. The results of an *in silico* evaluation of TEs detection and annotation tools indicate that their performance can be improved by using machine learning classifiers.

**Keywords:** Transposable Elements, Machine Learning, Genomics.

## 1 Introduction

Transposable Elements (TE), also known as transposons, are sequences of DNA that move and transpose within a genome. TE's role as mutation agents is important in both genome alteration diseases and on species evolution [3] [4] [12] [8][9][2]. Several methods have been developed to discover and annotate Transposable Elements. In [1] an extensive list of TE detection methods is surveyed. These methods have been classified in four main categories [1]: *De novo*; Structure-based; Comparative Genomic; and Homology-based. Although there are different tools, based on these methodologies, for detecting transposable elements there is not any single tool achieving good results on different types of TEs.

In this paper we evaluate existing TE detection tools using *in silico* data and study if Machine Learning techniques can be used to combine several TE detection tools predictions in order to improve the overall detection accuracy.

The remainder of the paper is organised as follows. Section 2 explains the data generation process for evaluation the TEs detection tools. In Section 3 we present an empirical evaluation of the TEs detection tools. In Section 4 we explain the ML experiments

M.S. Mohamad et al. (Eds.): *7th International Conference on PACBB*, AISC 222, pp. 85–91.
DOI: 10.1007/978-3-319-00578-2_12

to improve the performance of detecting TEs. Finally, in Section 5, we draw some conclusions.

## 2 *In Silico* Data

In order to evaluate the TE detection tools it is essential to have curated data sets of genome sequences. Table 1 summarizes the data used (namely TEs, genes, repetitive elements, etc) to assemble 'artificial' sequences. The set of 'real' genes was obtained from FlyBase [13] (Drosophila melanogaster). The 'real' TEs were obtained from Repbase [7] and from Gydb [11]. Other variables considered in the simulation were mutations, either point mutations or *indel* mutations, the length, composition and abundance of TE.

**Table 1.** Data used to produce simulated sequences

| Element type | | Number |
|---|---|---|
| TEs | Autonomous LTR Retrotransposons | 2248 |
| | Non autonomous LTR Retrotransposons | 379 |
| | DIRS | 14 |
| | Non-LTR Retrotransposons | 140 |
| | Autonomous non-LTR Retrotransposons | 604 |
| | Non autonomous non-LTR Retrotransposons | 384 |
| | TIR | 1247 |
| | DNA Transposons | 628 |
| | Helitrons | 139 |
| | Politrons | 24 |
| Genes | | 15458 |
| Repetitive Elements | | 147 |

The simulation parameters included the length of the sequence to be produce, percentage of genes included in the sequence in relation to its total length, percentage of TEs included in the sequence in relation to its total length, percentage of repetitive elements (no transposons included) that should be included in the sequence in relation to its total length. rate of insertions, deletions and replacements. Producing sequences using different combinations of these parameter's values allowed us to generate a diverse set of DNA sequences. The output data of a simulation is a set of sequences, written in FASTA [10] format, and an annotation file containing all TEs and repetitive elements locations inside each sequence. A simulated sequence consists of genes, transposons and other repetitive elements filled with random nucleotides in the gaps between them. The quantity of TEs, genes and repetitive elements are defined by the parameters referred above.

## 3 Evaluation of Transposon Detection Tools

Each TE detection tool analyzes all the sequences (of a given data set) and produces as a result the annotations of the TEs. The general accuracy was computed based on the predicted location of TEs and the "true" locations generated by the simulator.

In this study we have evaluated five tools that we next briefly describe.

PILER[1] [5] is a *de novo* TE detection tool that adopts a heuristic-based approach to *de novo* repeat annotation that exploits characteristic patterns of local alignments induced by certain classes of repeats. The PILER algorithm is designed to analyze assembled genomic regions and find only repeat families whose structure is characteristic of known subclasses of repetitive sequences.

BLAT [15] is a mRNA/DNA alignment tool. It uses an index of all non-overlapping K-mers in a given genome to find regions likely to be homologous to the query sequence. It performs an alignment between homologous regions and stitches together these aligned regions into larger alignments.

CENSOR[2] [6] was designed to identify and eliminate fragments of DNA sequences homologous to any chosen reference sequences. It uses BLAST to identify matches between input sequences and a reference library of known repetitive sequences. The length and number of gaps in both the query and library sequences are considered along with the length of the alignment in generating similarity scores. This tool reports the positions of the matching regions of the query sequence along with their classification.

RepeatMasker[3] [14] discovers repeats and removes them to prevent complications in downstream analysis sequence assembly and gene characterization. Identification of repeats by RepeatMasker is based entirely upon shared similarity between library repeat sequences and query sequences. The output of the program is a detailed annotation of the repeats that are present in the query sequence as well as a modified version of the query sequence in which all the annotated repeats have been masked.

LTR_Finder[4] [17] predicts locations and structure of full-length LTR retrotransposons accurately by considering common structural features. LTR_FINDER identifies full-length LTR element models in genomic sequence. This program reports possible LTR retrotransposons models at different confidence levels.

In Table 2 the average accuracy of each tool regarding each TE type is presented. LTR_Finder achieved poorer results in finding most types of TEs. Overall both Censor and RepeatMasker were the most accurate tools in finding different types of TEs.

## 4 Machine Learning to Improve TEs Detection Tools

Based on the experimental results of the TEs detection tools evaluation (Section 3) we have investigated if Machine Learning (ML) algorithms could improve TEs detection. We have used a two step process for TEs detection using ML: i) determine if a certain item (subsequence) in the sequence is or not a TE (TE detection); and ii) if the item has been classified as a TE then we determine its boundaries (TE annotation). The first step is concerned with the choice of the best tools to identify a TE with some given characteristics. The second step aims at choosing a tool that minimizes the error of an inferred TE boundary.

[1] http://www.drive5.com/piler/

[2] http://www.girinst.org/downloads/software/censor/

[3] http://www.repeatmasker.org/

[4] http://tlife.fudan.edu.cn/ltr_finder/

**Table 2.** Accuracy (%) per TE type

| **Tool** | **Aut. LTR Retrotransposons** | **Non Aut. LTR Retrotransposons** | **DIRS** | **Non-LTR Retrotransposons** | **Aut. Non-LTR Retrotransposons** | **Non Aut. Non-LTR Retrotransposons** | **TIRS** | **DNA TEs** | **Helitrons** | **Politrons** | **All** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **BLAT** | 26.69 | 18.12 | 2.72 | 23.12 | 19.68 | 37.69 | 20.46 | 13.67 | 21.92 | 11.14 | 19.61 |
| **Censor** | **61.49** | **82.68** | **81.43** | **71.1** | **74.02** | **68.45** | **78.85** | **82.9** | **52.13** | 20.86 | **67.38** |
| **LTR_Finder** | 0.17 | 0.22 | 0.1 | 0.02 | 0 | 0 | 0 | 0 | 0 | 0 | 0.05 |
| **PILER** | 0.51 | 38.05 | 36.33 | 37.46 | 46.6 | 10.24 | 28.27 | 25.63 | 41.94 | **23.56** | 28.66 |
| **Repeat Masker** | 51.66 | 58.1 | 31.1 | 43.88 | 51.71 | 55.63 | 57.66 | 57.14 | 42.23 | 4.62 | 45.28 |

The *Rapidminer* [5] software which uses *Weka* [16] algorithm implementations was used to build the classifiers. The algorithms considered: *Weka*'s implementation of Neural networks using 500 training cycles and 0.3 of learning rate; Bayes Network; Random Forest classifier to build an ensemble of decision trees; Decision Trees based on the C4.5 algorithm. The classifiers performance was estimated by measuring the accuracy in a 10 fold cross-validation procedure.

The classification of a potential TE candidate as a TE or not is a typical classification problem. In these terms, we used a data set containing 325000 examples, equally distributed in terms of TE types and in terms of being real TEs or false positives. The features used as the input for the models were the discretized TE length (using Equal-depth Binning in 50 categories), the TE type, the tool that made the prediction (FOUNDTOOL), and a IS_TE feature as the class. The IS_TE feature is a boolean which indicates whether a given example is or is not a TE. Table 3 shows the results obtained for the different ML algorithms considered. The best results were achieved with Decision Trees with an average accuracy of 98%, although the difference to Random Forest is not significantly different.

**Table 3.** TE detection: accuracy using different classification algorithms

| **Algorithm** | **Accuracy (%)** |
|---|---|
| Neural Network | 69.01 |
| Naive Bayes Net | 96.30 |
| Random Forest | 98.90 |
| Decision Trees | 98.92 |

[5] http://www.rapidminer.com/

The sensitivity of the tools for the level of mutations present in the sequences analyzed was also a theme that we wanted to clarify. The results (not shown) suggest that BLAT and PILER tools are influenced by mutations present in the DNA sequences. On the other hand, the performance of Censor, LTR_Finder and RepeatMasker were not affected significantly by the level of mutations.

**Finding the Best TE Annotation Tool.** Which tool minimizes the predicted location error for a given TE candidate? To answer this question we used a set of 129 198 examples of TE elements equally distributed between the different TE classes. "bestTool" is the class label and we have used the following features: TE type; set of tools that have detected the TE in step1; number of such tools that have detected the TE in step 1; and the class of the tools that have detected the TE in step 1. The bestTool feature is the name of the tool with the minimum location error.

We tested different model generation algorithms, all subjected to a 10 fold cross-validation process, to assess their performance. In Table 4 the results obtained with the different learner algorithms are compared. Again, the model with highest accuracy was produced with Decision Trees. Table 5 presents the confusion matrix of this model. This classifier has a high accuracy and can perform well with the tested artificial data. It is also worth to mention that the LTR_Finder tool was never used in this context as the location error performance of this tool is considerable lower than the others.

Applying machine learning to construct classifiers in the TE detection scope can further improve the accuracy of TE detection and annotation. In all the different problems, the approach that produced best results was Decision Trees (W-J48 Weka implementation).

**Table 4.** TE annotation: Classification algorithms model comparison. ZeroR measures the majority class percentage and is used as a base line value.

| **Algorithm** | **Accuracy (%)** |
|---|---|
| Ridor | 96.43 (0.10) |
| Naive Bayes Net | 96.37 (0.18) |
| Random Forest | 96.56 (0.14) |
| Decision Trees | 96.56 (0.14) |
| ZeroR | 76.55 |

## 5 Conclusions

In this paper we have assessed a set of computational tools for detecting Transposable Elements. The results obtained suggest that both Censor and RepeatMasker are the most accurate tools in detecting TEs. In a particular category, Politron TEs, the PILER tool obtained the best results. The LTR_Finder tool has achieved, by far, the worse results in this comparison with very low accuracy in the detection of TE. BLAT and RepeatMasker had some problems detecting DIR TEs. On the other hand, Censor scored exceptionally well in this TE category. Politron TEs were also a problem for

**Table 5.** TE annotation: confusion matrix for the Decision Trees model

| | True BLAT | True Censor | True LTR_Finder | True PILER | True RepeatMasker | Class Prediction |
|---|---|---|---|---|---|---|
| **Predicted BLAT** | 98902 | 0 | 0 | 0 | 0 | 100.0 % |
| **Predicted Censor** | 1911 | 23427 | 0 | 0 | 0 | 92.5 % |
| **Predicted LTR_Finder** | 1 | 0 | 0 | 0 | 4 | 0.0 % |
| **Predicted PILER** | 140 | 71 | 0 | 0 | 85 | 0.0 % |
| **Predicted RepeatMasker** | 834 | 1395 | 0 | 0 | 2428 | 52.1 % |
| **Class Recall** | 97.2 % | 94.1 % | 0.0 % | 0.0 % | 96.6 % | |

tools like RepeatMasker, Censor and BLAT. In this case, PILER performed especially well, outscoring all the other tools.

In terms of inference of TE boundaries, except for the LTR_Finder performance, all the tools performed acceptably well. The biggest issues occurred on the detection of the boundaries of Politron TEs and PILER had some trouble in detecting DIR TEs.

Using different TE tools' predictions from simulated data sets, we generated two classifiers that predict: i) if a given TE candidate is a TE or not, and ii) if it was a TE, predict which tool to use to minimize the boundaries error of that TE.

All in all, we presented evidence that ML models can be used to boost the detection and annotation of existing TE computational tools. Further research is needed to confirm the results in real data.

**Acknowledgments.** We would like to thank the informatics course "Master in Informatics and Computing Engineering" from Faculdade de Engenharia da Universidade do Porto for providing the conditions for the fulfillment of this work. This work was financed by the ERDF  European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT  Fundaçção para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within projects: FCOMP - 01-0124-FEDER-022701, PTDC/EME-PME/108308/2008 - (COMP- 01-0124-FEDER-010262), and FCOMP-01-0124-FEDER-008916 (PTDC/BIA-BEC/099933/2008).

## References

1. Bergman, C.M., Quesneville, H.: Discovering and detecting transposable elements in genome sequences. Briefings in Bioinformatics 8(6), 382–392 (2007)
2. Chénais, B., Caruso, A., Hiard, S., Casse, N.: The impact of transposable elements on eukaryotic genomes: From genome size increase to genetic adaptation to stressful environments. Gene (2012)
3. Casacuberta, E., Gonzlez, J.: The impact of transposable elements in environmental adaptation. Mol. Ecol. (2013)
4. Cowley, M., Oakey, R.J.: Transposable elements re-wire and fine-tune the transcriptome. PLoS Genet. 9(1) (2013)

5. Myers, E.W., Edgar, R.C.: PILER: identification and classification of genomic repeats. Bioinformatics 21, 152–158 (2005)
6. Jurka, J., Klonowski, P., Dagman, V., Pelton, P.: Censora program for identification and elimination of repetitive elements from DNA sequences. Computers & Chemistry 20(1), 119–121 (1996)
7. Jurka, J., Kapitonov, V.V., Pavlicek, A., Klonowski, P., Kohany, O., Walichiewicz, J.: Repbase update, a database of eukaryotic repetitive elements. Cytogentic and Genome Research 110, 462–467 (2005)
8. Kim, Y.J., Lee, J., Han, K.: Transposable elements: No more 'junk dna'. Genomics Inform. 10(4), 226–233 (2012)
9. Koso, H., Takeda, H., Yew, C.C., Ward, J.M., Nariai, N., Ueno, K., Nagasaki, M., Watanabe, S., Rust, A.G., Adams, D.J., Copeland, N.G., Jenkins, N.A.: Transposon mutagenesis identifies genes that transform neural stem cells into glioma-initiating cells. Proceedings of the National Academy of Sciences 109(44), E2998–E3007 (2012)
10. Pearson, W.R., Lipman, D.J.: Rapid and sensitive protein similarity searches. Science 227(4693), 1435–1441 (1985)
11. Llorns, C., Futami, R., Bezemer, D., Moya, A.: The ::::gypsy:::: Database (gydb) of mobile genetic elements. Nucleic Acids Research 36(Database-Issue), 38–46 (2008)
12. Lisch, D.: How important are transposons for plant evolution? Nat. Rev. Genet. 14(1), 49–61 (2013)
13. McQuilton, P., St. Pierre, E., Thurmond, J.: Flybase 101 - the basics of navigating flybase. Nucleic Acids Research 40(Database-Issue), 706–714 (2012)
14. Green, P., Smit, A.F.A., Hubley, R.: RepeatMasker Open-3.0
15. Kent, W.: Blat the blast-like alignment tool. Genome Research 12 (2002)
16. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann (2005)
17. Xu, Z., Wang, H.: LTR_FINDER: an efficient tool for the prediction of full-length LTR retrotransposons. Nucleic Acids Research 35(suppl. 2), W265–W268 (2007)