Improving the performance of Transposable Elements detection tools

Tiago Loureiro[1], Rui Camacho[1,2,*], Jorge Vieira[3] and Nuno A. Fonseca[4,5]

[1]DEI & Faculdade de Engenharia, Universidade do Porto, Portugal, http://www.fe.up.pt

[2]LIAAD-INESCTEC, Universidade do Porto, Portugal, http://www2.inescporto.pt

[3]IBMC - Instituto de Biologia Molecular e Celular & Universidade do Porto, Portugal, http://www.ibmc.up.pt

[4]European Molecular Biology Laboratory, European Bioinformatics Institute (EMBL-EBI), United Kingdom, http://www.ebi.ac.uk/

[5] CRACS-INESCTEC, Portugal, http://cracs.fc.up.pt/

Summary

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 tool 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 constructed classifiers.

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 quite important in both genome alteration diseases and in species evolution [1, 2, 3, 4, 5, 6]. Several methods have been developed to discover and annotate Transposable Elements. In [7] we can find an extensive survey of TE detection methods. These methods have been broadly classified in four main categories [7, 8]: *De novo*; Structure-based; Comparative Genomic; and Homology-based. Although there are different tools, based on the above mentioned methodologies, there is not any single tool achieving good results on different types of TEs. However the results of TE detection vary from tool to tool. TE detection agreement between different tools is one of the main problems, since tools may identify or not a given TE and even if identified they can disagree on its length and/or on its end-point's position within the genome.

*To whom correspondence should be addressed. Email: rcamacho@fe.up.pt

On another perspective, according to [7] and [8], integration of multiple approaches will further advance the computational analysis of this dynamic component. This means that integrating the best of each relevant tool can improve the overall detection accuracy and provide researchers better results in TE detection.

The aim of the work described in this paper is to assess the performance of existing TE detection tools. We have used simulated data sets of DNA sequences that have TE sequences present in *a priori* known positions. Using these data sets, several TE detection tools were evaluated by comparing the inferred predictions to the known locations of the TEs in the simulated data sets. Then, using the gathered data, machine learning (ML) techniques were used to create a model that combines the different methodologies with the aim to increase the accuracy of the detection and annotation of transposable elements.

The remainder of the paper is organized as follows. Section 2 provides a survey study and characterization of the TE detection methodologies and tools. Section 3 explains the data generation process for evaluating the TEs detection tools. In Section 4 we present an empirical evaluation of the TEs detection tools. In Section 5 we explain the ML experiments to improve the performance of detecting TEs. Finally, in Section 6, we draw some conclusions.

2 Transposable Elements detection methodologies and tools

In this study we have adopted the TE classification proposed by Bergman and Quesneville [7]:

- *De novo*: this approach looks for similar sequences found at multiple positions within a sequence;
- Homology-based: use known TEs to discover the TEs in a genomic sequence;
- Structure-based: this approach finds TEs using knowledge from their structure;
- Comparative genomics: compares genomes to find insertion regions which can be TEs or caused by TEs.

We have considered in the evaluation only the tools that are: i) publicly available and open source; ii) runnable from a command-line; and iii) supported by scientific studies. Table 1 lists and summarizes the TE detection tools used in this study.

BLAT [9] 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 [10] was designed to identify and eliminate fragments of DNA sequences homologous to any chosen reference sequences. It uses BLAST [14] (Basic Local Alignment Search Tool), a software package for rapid searching of nucleotide and protein databases, 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

Table 1: Transposable Elements detection tools used in the study. The number of citations was obtained from Google Scholar in 13/09/2013.

Name	Type	Operating System	URL	Citations
BLAT[9]	Homology - based	Unix and online	http://www.soe.ucsc.edu/ kent	3624
CENSOR[10]	Homology - based	Unix and online	http://www.girinst.org/downloads/software/censor/	329
LTR_Finder[11]	Structure - based	Unix	http://tlife.fudan.edu.cn/ltr_finder/	141
PILER[12]	*De novo*	Unix	http://www.drive5.com/piler/	164
RepeatMasker[13]	Homology - based	Unix and online	http://www.repeatmasker.org/	144

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.

LTR_Finder [11] predicts the location and structure of full-length LTR retrotransposons accurately by considering common structural features. LTR_Finder identifies full-length LTR element models in genomic sequences. This program reports possible LTR retrotransposons models at different confidence levels.

PILER [12] is a *de novo* TE detection tool that adopts a heuristic-based approach to 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 to find only repeat families whose structure is characteristic of known sub-classes of repetitive sequences. It works on the premise that the entire DNA sequence is assembled with a reasonably low number of errors because the algorithm is completely dependent on the position of repeats in the genome for all classification.

RepeatMasker [13] discovers repeats and removes them to prevent difficulties in downstream analysis sequence assembly and gene characterization. Identification of repeats by RepeatMasker is based entirely upon the 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.

3 *In silico* data

In order to evaluate the TE detection tools it is essential to have curated data sets (experimental or simulated) of genome sequences with transposable elements. The curated data sets were generated *in silico*. The aim of the simulation procedure was to obtain sequences with a random number of TEs in random positions. The simulation parameters included: the length of the sequence to be produced; the percentage of genes included in the sequence in relation to its total length (genes %); the percentage of TEs included in the sequence in relation to its total length (TEs %); the percentage of repetitive elements (no transposons included) included in the sequence in relation to its total length (Repetitive elements %); and the number of mutations (insertions, deletions and replacements) per 1000 nucleotides (mutations %).

Table 2: 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

Producing sequences using different combinations of these parameters' values allowed us to generate a diverse set of DNA sequences. The output data of a simulation is a file with a set of sequences, written in FASTA [15] format, and an annotation file containing all TEs and their 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.

Table 2 summarizes the data (namely TEs, genes, repetitive elements, etc) used to assemble 'artificial' sequences. The set of 'real' genes was obtained from FlyBase [16] (*Drosophila melanogaster*). The 'real' TEs were obtained from Repbase [17] and from Gydb [18]. Other variables considered in the simulation were mutations, either point mutations or *indel* mutations, the length, composition and abundance of TEs.

4 Evaluation of Transposon detection tools

In this section we first present the global results comparing the five tools under assessment and then we present the results of a detailed study of sensitivity of the tool with best global performance.

4.1 Global comparison

Each TE detection tool analyzes all sequences (of a given data set) and produces as a result the annotations of the TEs. The general accuracy[1] was computed based on the predicted location of TEs and the "true" locations generated by the simulator.

[1]The evaluation measures used include: Accuracy=$\frac{TP+TN}{TP+FP+TN+FN}$; Precision=$\frac{TP}{TP+FP}$; and Recall=$\frac{TP}{TP+FN}$, where TP, FP, TN, FN are, respectively, the number of true positives, false positives, true negatives and false negatives as predicted by the tool.

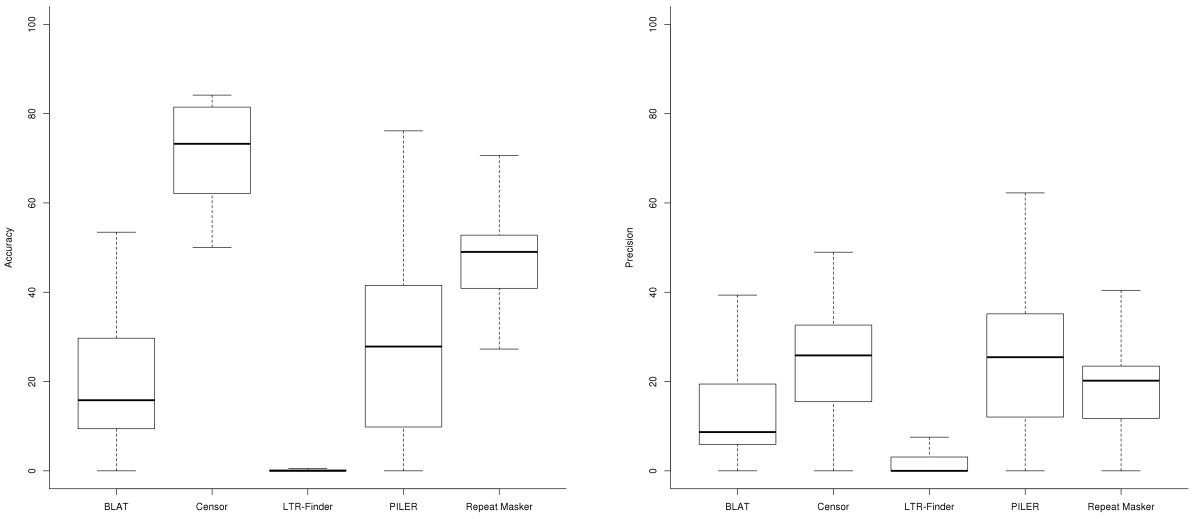

Figure 1: Accuracy and precision of each tool on all data sets.

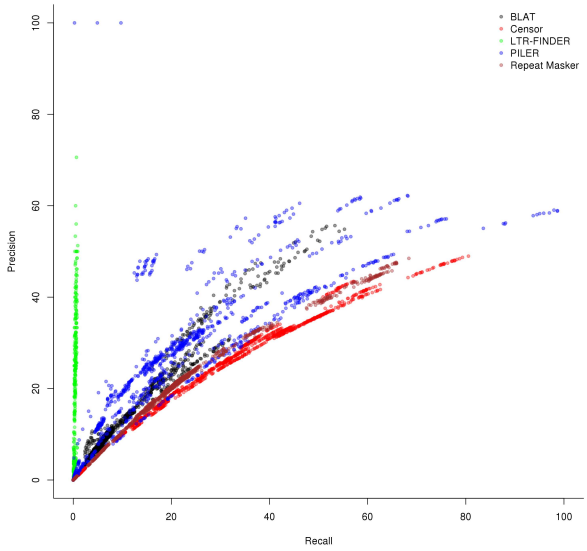

Figure 2: Precision and recall for all tools on all data sets.

Figure 1 and Figure 2 show the accuracy, precision and recall of all tools across all data sets. The accuracy of each tool on the different types of TEs is shown in Table 3. LTR_Finder had poor results with an average accuracy bellow 1%. The higher accuracy of all the tools was achieved by Censor with an average accuracy above 70%. RepeatMasker had also interesting results, with an average accuracy of roughly 50%. The precision of RepeatMasker and Censor is comparable to PILER. Overall, both Censor and RepeatMasker were the most accurate tools in finding different types of TEs.

4.2 Individual sensitivity results

We have made an extensive set of sensitivity studies on the Censor tool, the one with overall best accuracy, precision and recall. The results of Censor assessment are summarized in Figure 3. It shows the accuracy of Censor segmented by the different input parameters used to generate the simulated data sets, such as amount of TEs, genes, repetitive elements and mutations.

There are minor but not significant differences in Censor's accuracy when analyzing sequences with 0, 1 or 2 percent of *indel* and point mutations. The remaining three parameters, the amount of genes, repetitive elements and TEs in the simulated sequences did not affect significantly the accuracy of Censor.

Censor's detection accuracy (Table 3) for all but Politron TEs is relatively good as it ranges from more than 50% to over than 80% in DIRs, DNA transposons and non autonomous LTR retrotransposons. Politron detection using this tool is much weaker than with the other classes as the accuracy of this tool in this TE category is roughly above 20%.

Table 3: 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

5 Machine Learning to improve TEs detection tools

Based on the experimental results of the TEs detection tools evaluation (Section 4) 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 (sub-sequence) in the sequence is or is 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 the classifier that minimizes the error of an inferred TE boundary.

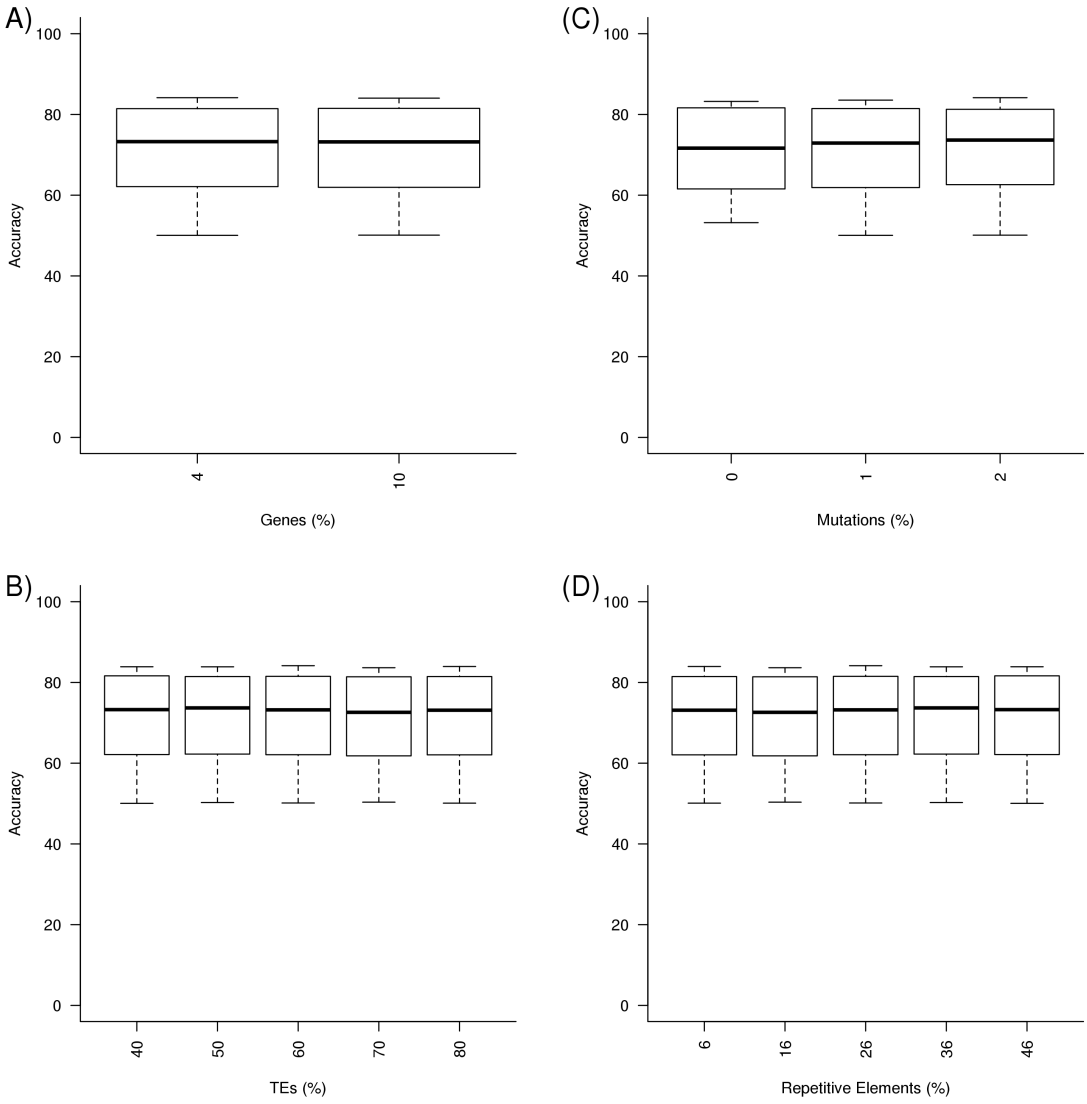

Figure 3: Censor's accuracy for different parameters used in the generation of data sets: A) genes (%); B) mutations (%) C) TEs (%); and D) repetitive elements (%).

The *Rapidminer* [2] software which uses *Weka* [19] algorithm implementations was used to build the classifiers. The algorithms considered: *Weka*'s implementation of Neural networks; Bayes Network; Random Forest classifier to build an ensemble of decision trees; Decision Trees based on the C4.5 algorithm. Different values of the algorithms's parameters were experimented and the results reported were obtained with the combinations giving the best results. 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: i) discretized TE length (using Equal-depth Binning in 50 categories); ii) the TE type; iii) the tool that made the prediction (FOUNDTOOL); and iv) a IS_TE feature as the

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

Table 4: 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

class. The IS_TE feature is a boolean feature which indicates whether a given example is or is not a TE. Table 4 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 significant.

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.

5.1 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[3] 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 5 the results obtained with the different learner algorithms are compared. Again, the model with highest accuracy was produced with Decision Trees. Table 6 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 considerably 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).

[3]Number of cases where the existence of a TE was correctly predicted.

Table 5: 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

Table 6: 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 %	

6 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 tools like RepeatMasker, Censor and BLAT. In this case, PILER performed 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.

Acknowledgements

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 Program (operational program for competitiveness) and by National Funds through the FCT – Fundação 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] E. Casacuberta and J. González. The impact of transposable elements in environmental adaptation. *Molecular Ecology*, 22(6):1503–1517, 2013.

[2] M. Cowley and R. Oakey. Transposable elements re-wire and fine-tune the transcriptome. *PLoS Genetics*, 9(1):e1003234, 2013.

[3] D. Lisch. How important are transposons for plant evolution? *Nature Reviews Genetics*, 14(1):49–61, 2013.

[4] Y. Kim, J. Lee and K. Han. Transposable Elements: No More 'Junk DNA'. *Genomics & Informatics*, 10(4):226–233, 2012.

[5] H. Koso, H. Takeda, C. C. Yew et al. Transposon mutagenesis identifies genes that transform neural stem cells into glioma-initiating cells. *Proceedings of the National Academy of Sciences U.S.A.*, 109(44):E2998–E3007, 2012.

[6] B. Chénais, A. Caruso, S. Hiard and N. Casse. The impact of transposable elements on eukaryotic genomes: From genome size increase to genetic adaptation to stressful environments. *Gene*, 509(1):7–15, 2012.

[7] C. M. Bergman and H. Quesneville. Discovering and detecting transposable elements in genome sequences. *Briefings in Bioinformatics*, 8(6):382–392, 2007.

[8] M. J. Curcio and K. M. Derbyshire. The outs and ins of transposition: from Mu to Kangaroo. *Nature Reviews Molecular Cell Biology*, 4(11):865–877, 2003.

[9] W. Kent. BLAT – the BLAST-like alignment tool. *Genome Research*, 12(4):656–664, 2002.

[10] J. Jurka, P. Klonowski, V. Dagman and P. Pelton. Censor—a program for identification and elimination of repetitive elements from DNA sequences. *Computers & Chemistry*, 20(1):119–121, 1996.

[11] Z. Xu and H. Wang. LTR_FINDER: an efficient tool for the prediction of full-length LTR retrotransposons. *Nucleic Acids Research*, 35(suppl 2):W265–W268, 2007.

[12] R. Edgar and E. Myers. PILER: identification and classification of genomic repeats. *Bioinformatics*, 21(suppl 1):i152–i158, 2005.

[13] A. F. Smit and P. Green. Repeatmasker. *Published on the web at http://www.repeatmasker.org*, 1996.

[14] I. Korf, M. Yandell and J. Bedell. *BLAST*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2003.

[15] D. Lipman and W. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, 1985.

[16] P. McQuilton, S. E. S. Pierre and J. Thurmond. FlyBase 101 – the basics of navigating FlyBase. *Nucleic Acids Research*, 40(D1):D706–D714, 2012.

[17] J. Jurka, V. Kapitonov, A. Pavlicek, P. Klonowski, O. Kohany and J. Walichiewicz. Repbase Update, a database of eukaryotic repetitive elements. *Cytogentic and Genome Research*, 110(1-4):462–467, 2005.

[18] C. Lloréns, R. Futami, D. Bezemer and A. Moya. The Gypsy Database (GyDB) of mobile genetic elements. *Nucleic Acids Research*, 36(suppl 1):D38–D46, 2008.

[19] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2nd edition edition, 2005.