BIOINFORMATICS ORIGINAL PAPER

Vol. 25 no. 23 2009, pages 3093–3098
doi:10.1093/bioinformatics/btp552

Phylogenetics

Reproducing the manual annotation of multiple sequence alignments using a SVM classifier

Christian Blouin[1,2,3,*], Scott Perry[2], Allan Lavell[2], Edward Susko[3,4] and Andrew J. Roger[1,3]

[1]Department of Biochemistry and Molecular Biology, Dalhousie University, Sir Charles Tupper Medical Building, Halifax NS B3H 1X5, [2]Faculty of Computer Science, Dalhousie University, Halifax NS B3H 5W1, [3]Centre for Genomics and Evolutionary Bioinformatics, Dalhousie University and [4]Department of Mathematics and Statistics, Dalhousie University, Halifax NS B3H 6J3, Canada

Received on July 28, 2009; revised on August 31, 2009; accepted on September 16, 2009

Advance Access publication September 21, 2009

Associate Editor: Martin Bishop

ABSTRACT

Motivation: Aligning protein sequences with the best possible accuracy requires sophisticated algorithms. Since the optimal alignment is not guaranteed to be the correct one, it is expected that even the best alignment will contain sites that do not respect the assumption of positional homology. Because formulating rules to identify these sites is difficult, it is common practice to manually remove them. Although considered necessary in some cases, manual editing is time consuming and not reproducible. We present here an automated editing method based on the classification of 'valid' and 'invalid' sites.

Results: A support vector machine (SVM) classifier is trained to reproduce the decisions made during manual editing with an accuracy of 95.0%. This implies that manual editing can be made reproducible and applied to large-scale analyses. We further demonstrate that it is possible to retrain/extend the training of the classifier by providing examples of multiple sequence alignment (MSA) annotation. Near optimal training can be achieved with only 1000 annotated sites, or roughly three samples of protein sequence alignments.

Availability: This method is implemented in the software MANUEL, licensed under the GPL. A web-based application for single and batch job is available at http://fester.cs.dal.ca/manuel.

Contact: cblouin@cs.dal.ca

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Sequences showing a significant similarity are assumed to be homologous. In fact, significant similarity is commonly used as a basis to assemble datasets; the most common strategies use the BLAST family of algorithms (Altschul *et al.*, 1990). The exact algorithm to align k sequences of n sites has a prohibitive computational complexity, even with a small number of sequences (Thompson *et al.*, 1999). Many heuristics are based on the fundamental algorithms for the global (Needleman and Wunsch, 1970) or the local (Smith and Waterman, 1981) pairwise sequence alignment. The most famous strategy is the progressive alignment method (Feng and Doolittle, 1987), which is implemented in popular packages such as Clustalw (Thompson *et al.*, 1994), MUSCLE (Edgar, 2004) and T-Coffee (Notredame *et al.*, 2000), to name a few. Radically different approaches use variants of hidden Markov models as in HMMER (Eddy, 1998), Probcons (Do *et al.*, 2005) and fast statistical alignment (FSA; Bradley *et al.*, 2009). Obtaining biologically accurate alignments for more than two sequences remains a challenge (Notredame, 2007), as even the best methods can fail to readily align conserved motifs (Edgar, 2004).

In an accurate multiple sequence alignment (MSA), each character state in a site (column) is homologous (i.e. all character states in a column evolved through vertical descent from a common ancestral character). Making the assumption of homology is important because it allows to relate characters through an underlying evolutionary process. There is an extensive documentation on the impact of alignment quality on phylogeny (Landan and Graur, 2009; Löytynoja and Goldman, 2008; Ogdenw and Rosenberg, 2006; Wong *et al.*, 2008). Insertions and deletions in sequences makes the alignment process difficult (Nuin *et al.*, 2006). The main issue is that MSA algorithms attempt to minimize the number of gaps, resulting in optimal alignments that are shorter than the correct alignment (Landan and Graur, 2009) due to 'collapsed-insertions' (Löytynoja and Goldman, 2008; Lunter *et al.*, 2008), gap attraction (Lunter *et al.*, 2008) and gap wandering (Holmes and Durbin, 1998; Lunter *et al.*, 2008). For this reason, an optimal MSA can contain sites where the assumption of positional homology does not hold. In this work, we refer to these sites as *invalid*. The existence of a structural alignment can assist in determining positional homology. BaliBASE (Thompson *et al.*, 2005), SABmark (Van Walle *et al.*, 2004) and the PREFAB benchmark (Edgar, 2004) are three standard alignments datasets which use this criterion to determine the correct alignments. In practice, comparative structural data is often unavailable for a given family and thus cannot be consistently used. Many artifacts and bias can be found in MSAs and the validity of a site MSA is difficult to determine. For brevity, this work will refer to *possibly* valid sites as valid.

[*]To whom correspondence should be addressed.

The quality of an alignment is a function of the validity of its sites. A case can be made that this quality should be accessed using rigorous methods (Lassmann and Sonnhammer, 2005; Lunter *et al.*, 2008). Some alignment methods provide an intrinsic evaluation of site-wise alignment quality (Bradley *et al.*, 2009; Do *et al.*, 2005; Lassmann and Sonnhammer, 2007). Another method is the head-or-tail (HoT) method (Landan and Graur, 2007) which was demonstrated to perform well (Hall, 2008) and is independent of the alignment method used to prepare an MSA.

To improve the signal-to-noise ratio in an MSA, masking or removing invalid sites is common practice. It is difficult to classify invalid sites using deterministic rules with an acceptable balance of specificity and sensitivity. MSAs are therefore often annotated manually with expert judgment. This method is not repeatable and does not scale to large numbers of alignments. The curation of alignments can be performed using existing methods such as GBLOCKS (Castresana, 2000) or AL2CO (Pei and Grishin, 2001). GBLOCKS is a program that is designed to take a multiple protein sequence alignment as input and perform editing to produce a similarly formatted output with the invalid sites removed. An example where GBLOCKS was used to perform this task was in the curation of a set of 22 437 MSAs (Beiko *et al.*, 2005). While GBLOCKS can be used as an alignment editing method, and was shown to yield improved results for phylogenetic analysis (Castresana, 2000), it does not emulate the manual editing process. This approach effectively removes columns corresponding to the highest site rates (SRs) since they potentially contain multiple substitutions. However, these may be valid homologous sites that happen to be fast-evolving, and deleting them may remove valuable phylogenetic information about closely related sequences. In the AL2CO implementation, the concept of conservation index (CI) was introduced and recommended for use as a parameter for the refinement of multiple sequence alignments (Pei and Grishin, 2001). Treating AL2CO as an MSA editor requires a systematic method to select a CI threshold. It is difficult to determine what this threshold should be in practical settings.

In this work, we refer to editing as the process of masking and then removing entire sites. We introduce a simple and highly effective machine learning approach to capture the intrinsic rules of manual MSA editing. Editing is thus formulated as the binary classification of sites as valid or invalid using support vector machines (SVM). The classification of each site is used to apply a mask to an MSA. The raw data upon which the SVM classification is based on the quality of individual sites as a vector of features made of numerical values. The following sections present the details of this modeling process and compare the editing performance of our methods, MANUEL, with respect to two existing methods that are often used to edit MSAs. Finally, we demonstrate that the SVM classifier can learn from a relatively small quantity of training data. MANUEL thus can be tailored by researchers by simply training new classifiers from a small number of editing examples.

2 METHODS

2.1 Datasets

Thirty-eight ‘seed’ multiple sequence alignments were arbitrarily retrieved from PFAM (Finn *et al.*, 2008). A total of 17 934 sites of multiple sequence alignments were manually annotated by two of the authors (A.R. and C.B.). Two classes were identified during manual annotation: valid and invalid sites. Sites were classified as valid if there were no reasons to suspect that the site contained alignment artifacts. The average distribution of valid sites is 73.7% (per alignment min: 37.6%, max: 99.6%). In this work, this set of annotated sequences is referred to as the MANUEL corpus.

2.2 Parameterization

2.2.1 Modeling MSA editing Each site i was encoded as a feature vector $f_i = \{g_i, \mathrm{NSLR}_i, \mathrm{SR}_i, \mathrm{nPC}_i\}$. This encoding was derived from the definition of features that are related to qualitative properties considered during manual editing.

Gap ratio (g): this ratio expresses how many of the site is populated by non-gap character states. A non-zero gap ratio is a common trigger to consider the validity of a site and its neighbors. The feature g for site i is computed where C_i is the number of gap characters in i and N is the number of sequences in the alignment.

$$g_i = \frac{C_i}{N} \tag{1}$$

Normalized Site Likelihood Ratio (NSLR): this feature attempts to capture the tree-like signal in a site by comparing its likelihood assuming a reasonable tree topology ($\log(l_i)$) and the star tree ($\log(r_i)$) with infinite edge lengths. The reasonable tree is inferred using the Neighbor-joining (NJ) method (Saitou and Nei, 1987) on the unedited alignment. The likelihood of a site under the star-tree assumption is the product of the frequencies of all observed characters in the site. The normalization uncouples the site likelihood from the number of sequences in an alignment, and compensates for the presence of gap characters at site i.

$$\mathrm{NSLR}_i = \frac{\log(l_i) - \log(r_i)}{(1-g_i)N} \tag{2}$$

SR: this feature captures the relative rate of evolution of a site. SR(i) correspond to the rate of one of four equiprobable rate categories that contributes the most to the likelihood of site i under the JTT+Γ model. The rationale behind this feature is that invalid sites are more likely to appear fast evolving. The NSLR_i and SR_i are computed using the NJ tree topology, using a routine from the Bio++ library (Dutheil *et al.*, 2006) and the JTT matrix (Jones *et al.*, 1992).

Normalized Parsimony Count (nPC): this feature attempts to capture the plausibility of the gap opening/closing pattern. Each character is first re-encoded as either a residue or a gap character. The parsimony count for each site is then computed using the topology of the NJ tree. This parameter is normalized with respect to N.

Neighborhood: an important factor to determine whether a site is valid during the manual editing process is the validity of neighboring sites. We captured this by classifying the middle site of a window of three consecutive sites. Thus, a site i is encoded as the concatenation of features f_i' such that $f_i' = \{f_{i-1}, f_i, f_{i+1}\}$. The feature vector for f_0 and f_{N+1} is set to $\{0,0,0,0\}$. This results in the classification of i using 12 instead of 4 features.

2.2.2 SVM classification Cross-validation on the entire training corpus was performed on one dataset at a time. During cross-validation, a model is trained on all but one alignment from the training corpus and testing is performed on the withheld data. This is roughly the equivalent of evaluating the performance of the classifier with a 38-fold cross-validation

2.2.3 SVM implementation LibSVM package (Chang and Lin, 2001) was employed to build our application. We used the Python interface for this library. The implementation of this method is simple and can be reproduced wherever there is support for a SVM library.

2.3 ROC analysis

The receiver operating characteristics (ROC) (Fawcett, 2006) analysis was performed with the ROCR package (Sing *et al.*, 2005) for the R statistical environment (R Development Team, 2009).

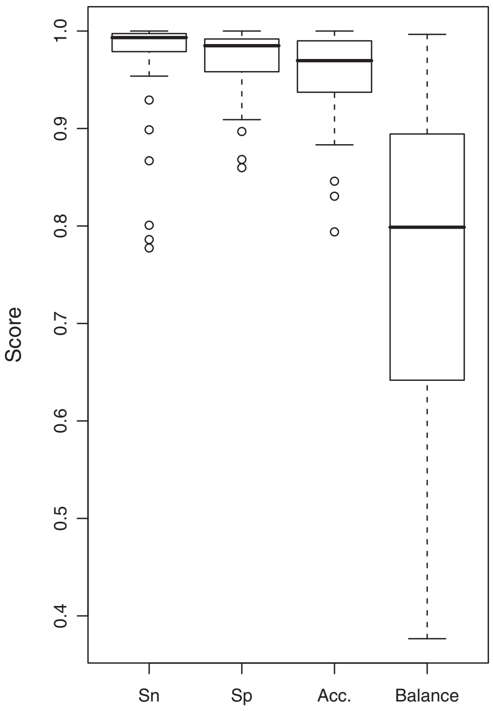

Fig. 1. Boxplot of the classification performance for single alignments in the MANUEL corpus. The SVM model for a given alignment *A* included all other alignments in MANUEL other than A. Sn, Sp and Acc stand, respectively, for sensitivity, specificity and accuracy. The proportion of valid sites is shown as the balance of the dataset for reference. The balance corresponds to the specificity and accuracy of a baseline classifier which considers all sites as valid. The sensitivity of this classifier would be 1.0.

3 RESULTS

3.1 Performance of the classifier

The editing of a MSA is the application of a mask generated by the annotation of all sites as either valid or invalid. This annotation is formulated as a classification problem where the classifier must identify valid sites in sequence alignments. A true positive (TP) is a valid site classified as such, a false positive (FP) is an invalid site classified as valid, while a false negative (FN) is a valid site classified as invalid. The accuracy of this classification is the number of sites classified as manually annotated. Sensitivity (Sn) and specificity (Sp) were computed as follows:

$$\mathrm{Sn}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{3}$$

$$\mathrm{Sp}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{4}$$

A sensitive classifier preserves as many valid sites as possible while a specific classifier minimizes the number of invalid sites in the final alignment.

Figure 1 shows the statistics of classification performance on individual alignments. These performances were obtained using a per-dataset cross-validation procedure as described in Section 2. A complete breakdown of these values is available as the Supplementary Material. Table 1 reports the classification performances for a single site and for the middle site in a window of 3. Although the process of manual annotation considers the neighborhood of a site, the classification of the middle site of a window of 3 is less accurate than a single site classification. The benefit of considering the neighborhood is offset by the cost of training in a higher dimensional space.

Table 1. Cross-validated performances of the classification of valid sites

Experiment	Sn	Sp	Accuracy
f_i	0.967	0.967	0.950
f_i'	0.984	0.911	0.917
GBLOCK[a]	0.431	0.999	0.584
GBLOCK[b]	0.520	0.993	0.647
GBLOCK[c]	0.544	0.963	0.651
GBLOCK[d]	0.909	0.735	0.694

SVM classification on single sites f_i and window of $3f_i'$. The performance of GBLOCKS was evaluated under four sets of parameters.

[a] Min. block length = 10, no gaps (Default).

[b] Min. block length = 5, $\frac{1}{2}$ gaps.

[c] Min. block length = 2, all gaps.

[d] Min. block length = 2, all gaps, < 32K non-conserved contiguous positions.

Table 2. Fraction of sites classified as valid in BaliBASE alignments

Reference set	Test set	Valid (%)
Ref1	All	81.9
Ref1	Test1	82.7
Ref1	Test2	77.9
Ref1	Test3	83.3
Ref2	All	78.7
Ref3	All	67.3
Ref3	Test	67.6
Ref3	Test1	67.1
Ref4	All	26.1
Ref5	All	58.9

Includes only MSA with five or more sequences.

There are more valid than invalid sites in a typical MSA. For this reason, the performance reported in this study should be compared with the performance of a trivial classifier which annotates all sites as valid. The accuracy of this trivial classifier on the cross-validated corpus would be 73.7%.

3.2 Editing BaliBASE alignments

All alignments from BaliBASE were edited using MANUEL. The proportion of sites classified as valid is shown in Table 2. Since all sites in BaliBASE are assumed to be correct, one could have expected to preserve all sites in this benchmark: this is not the case. The actual annotation of these MSAs is provided as Supplementary Material.

3.3 Comparison with other systems

As mentioned previously in Section 1, tools exist that can be used to assist in automatically editing sequence alignments such as GBLOCKS and AL2CO. Although these methods are not explicitly designed to perform this task, their performance can be used as a baseline. GBLOCKS was treated as a sequence annotator by

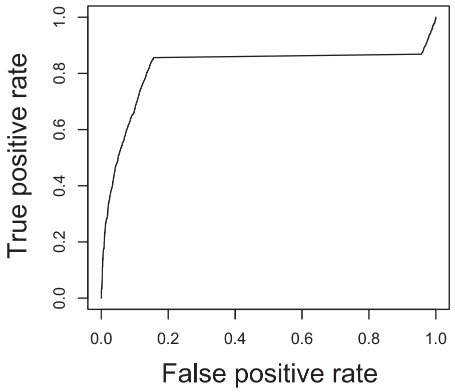

Fig. 2. ROC analysis of the conservation scores from AL2CO as a classifier for sequence editing. All sites from the MANUEL corpus were scored using AL2CO's default arguments, These scores were then compared against the corpus' manual annotation.

processing MSA under four sets of input parameters. The details of these input parameters are described in Table 1. By default, GBLOCKS provides very stringent editing which can be relaxed to improve the sensitivity and accuracy. The highest accuracy was obtained with the most permissive set of parameters, and achieved 69%. This accuracy is in fact less than the accuracy expected by the trivial classifier.

It is possible to use scoring schemes and a cutoff to devise an MSA editing strategy. The main problem in this case is to determine what this cutoff value should be, and demonstrate that this cutoff is appropriate for all MSAs that are to be edited. Figure 2 explores the performance of a cutoff-based strategy using the CI derived from AL2CO. AL2CO processed all MSA from the MANUEL corpus with the application's default value. The validity of sites was difficult to resolve using a cutoff approach because the distribution of CIs is located near the minimum (Supplementary Fig. 1). In practice, using AL2CO is difficult because it is impossible to determine the optimal CI cutoff without manually annotating the alignment.

3.4 Stability of the classifier

This system was tested on different combinations of SVM parameters (C, γ and kernel type). The kernel type selected is the Radial Basis Function (RBF). Preliminary tests have demonstrated that the classification accuracy is near optimal for a wide range of the RBF kernel's C and γ parameters. This indicates that the parameters are not dataset dependent and precise values are not critical to the quality of the classification. The values used for the RBF kernel were selected to be 2.0 for C and 8.0 for γ.

3.5 Accuracy versus training set size

Typically, the main limitation to deploying and using machine learning systems is the requirement for annotated training sets. Preparing these sets can be a time consuming task and often is not trivial to complete. Our system provides the functionalities to simplify this annotation procedure. Figure 3 provides an appreciation of the quantity of data that is required to obtain reasonable classification results. Training sets of fixed sizes were generated by random sampling of sites from the MANUEL corpus. An additional 3000 sites not included in the training set were randomly selected to be used for testing. The performance of classification is near optimal with 1000 or more sample sites. This implies that only a few alignments need to be provided as examples of manual editing to properly train the classifier.

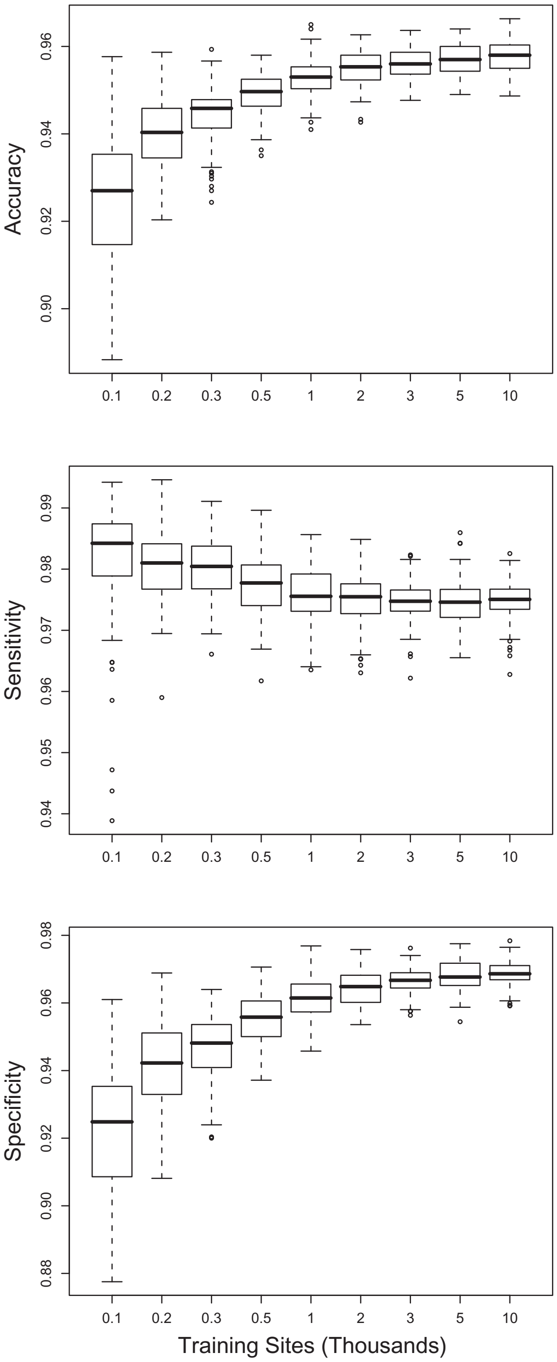

Fig. 3. Boxplot of the classification performance of 3000 sites with respect to the size of the training set. Training set sizes are expressed in thousands of sites. A subset of the MANUEL corpus was randomly selected as a training set, while another subset of 3000 sites was selected for testing. Each training set size category was evaluated with 100 replicate experiments.

The MANUEL implementation trains by comparing raw and edited alignments, subsequently annotating them so they can be added to the existing training corpus. The key issue to consider is whether the training set is a representative set of MSAs. Consideration must be given to providing examples with a representative balance of valid/invalid sites. Experiments that are not shown here indicate that the four features provide somewhat redundant signal to the classifier. However, all four parameters are used to provide a more diverse source of signal to accommodate different types of data.

4 DISCUSSION

The results in Figure 1 indicate that, with a window size of 1, the SVM classifier trained on the MANUEL corpus v1.0 can reproduce the manual editing of alignments with an accuracy of 95.0%. The errors introduced in the editing involve an approximatively equal number of FP and FN. Considering the immediate neighborhood of a site improves the sensitivity by classifying more sites as valid. This results in an overall decrease of performance in the accuracy of editing. This is probably due to the 3-fold increase in the size of the feature vectors. In evaluating the performance of the classifier, it is important to remember that this type of data is not naturally balanced as there are usually more valid than invalid sites in a typical alignment. However, the important result is that MANUEL clearly achieves a high accuracy in reproducing manual editing. Attempting to improve these performances is both unnecessary and probably impossible: the self-consistency of manual MSA annotation is unlikely to be 100%, although it is expected to be very close to it. The MANUEL corpus was annotated by two experts who did not consult on their criteria for the validity of a site. Despite these issues, it is clear that the classification of valid sites, using these features as input, is an easy problem for a SVM classifier. The objective of GBLOCKS is not exactly to edit alignments, but rather to create blocks of highly reliable alignments. GBLOCK does well for this purpose, but compares unfavorably as a sensitive MSA preprocessing tool. AL2CO can also be used as a way to filter the least conserved sites. The main problem AL2CO has in performing this task is to determine what the cutoff should be. This is made more difficult by the distribution of AL2CO conservation indices being heavily biased to the lower part of the range of values.

The deletion of sites from the BaliBASE benchmark clearly indicates that some sites assumed to be valid will be classified as invalid. MANUEL is not trained to identify ultimately correct sites as the information used to assert correctness cannot be extracted from sequence information alone. It is important to note that a classifier trained to edit MSAs is not capable to outperform the manual annotation of alignments (although it reproduces it with a high accuracy).

We have tested many types of classifiers for this task (Shan *et al.*, 2003) and subsequently with a number of artificial neural network architectures. SVMs prove to be robust and, by far, the most simple classifier to train and operate. SVMs are increasingly popular in bioinformatics. For example, an SVM was recently used to classify alignments which contain strongly supported discordant branches from alignment properties alone (Roettger *et al.*, 2009). It would be interesting to determine the sensitivity of the SVM classifier to the topology of the guide tree, and whether this sensitivity could be used to further discriminate sites that were involved in events of gene conversion.

It is important to note that, because MSA editing appears to be an easy problem for a SVM classifier, it is possible to refine or completely retrain the classifier to match specific annotation practices. Figure 3 demonstrates that reasonable performance can be achieved with a training set of about 1000 sites of raw alignments. For convenience, the MANUEL software automatically annotates and prepares the examples if provided with a copy of the raw alignment and its manually edited version. Because the parameters of the RBF kernel are stable, there is no reason to believe that C and γ should be adjusted, even if the training set is replaced or drastically changed. If in doubt, it would be advisable to test this assertion using a simple grid search for these two parameters. Finally, the intermediate feature file is kept so it is possible to add or remove features. It is possible to completely replace the parameterization routine without changing the classification code.

5 CONCLUSIONS

Reproducing the manual editing of multiple sequence alignments has two aims: (i) to automate the process to improve the quality of the input data for large-scale phylogenetic studies, and (ii) to improve the repeatability of this procedure. No claims are made about the objectivity of the editing process since our system is designed to reproduce the outcome of manual editing. To facilitate this, it is possible to train MANUEL simply by providing examples. This method makes it possible to apply a manual-quality alignment editing to large-scale phylogenetic studies.

ACKNOWLEDGEMENTS

We thank Wynne Lok, Aditya Aggarwal and Jiantao Lu for their early contribution. Further thanks go to Dr Robert Beiko and Haibin Liu for insightful discussions.

Funding: National Science and Engineering Research Council of Canada (Discovery Grant DGP 298397-04 to C.B.).

Conflict of Interest: none declared.

REFERENCES

Altschul,S.F. *et al.* (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Beiko,R.G. *et al.* (2005) Highways of gene sharing in prokaryotes. *Proc. Natl Acad. Sci. USA*, **102**, 14332–14337.

Bradley,R.K. *et al.* (2009) Fast statistical alignment. *PLoS Comput. Biol.*, **5**, e1000392.

Castresana,J. (2000) Selection of conserved blocks from multiple alignments for their use in phylogenetic analysis. *Mol. Biol. Evol.*, **17**, 540–552.

Chang,C.-C. and Lin,C.-J. (2001) LIBSVM: a library for support vector machines. Available at http://www.csie.ntu.edu.tw/cjlin/libsvm (last accessed date October 1, 2009).

Do,C.B. *et al.* (2005) Probcons: probabilistic consistency-based multiple sequence alignment. *Genome Res.*, **15**, 330–340.

Dutheil,J. *et al.* (2006) Bio++: a set of c++ libraries for sequence analysis, phylogenetics, molecular evolution and population genetics. *BMC Bioinformatics*, **7**, 188.

Eddy,S.R. (1998) Profile hidden Markov models. *Bioinformatics*, **14**, 755–763.

Edgar,R.C. (2004) Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.*, **32**, 1792–1797.

Fawcett,T. (2006) An introduction to ROC analysis. *Pattern Recogn. Lett.*, **27**, 861–874.

Feng,D.F. and Doolittle,R.F. (1987) Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.*, **25**, 351–360.

Finn,R.D. *et al.* (2008) The pfam protein families database. *Nucleic Acids Res.*, **36**, D281–D288.

Hall,B.G. (2008) How well does the hot score reflect sequence alignment accuracy? *Mol. Biol. Evol.*, **25**, 1576–1580.

Holmes,I. and Durbin,R. (1998) Dynamic programming alignment accuracy. *J. Comput. Biol.*, **5**, 493–504.

Jones,D.T. *et al.* (1992) The rapid generation of mutation data matrices from protein sequences. *Comput. Appl. Biosci.*, **8**, 275–282.

Landan,G. and Graur,D. (2007) Heads or tails: a simple reliability check for multiple sequence alignments. *Mol. Biol. Evol.*, **24**, 1380–1383.

Landan,G. and Graur,D. (2009) Characterization of pairwise and multiple sequence alignment errors. *Gene*, **441**, 141–147.

Lassmann,T. and Sonnhammer,E.L. (2007) Automatic extraction of reliable regions from multiple sequence alignments. *BMC Bioinformatics*, **8** (Suppl. 5), S9.

Lassmann,T. and Sonnhammer,E.L.L. (2005) Automatic assessment of alignment quality. *Nucleic Acids Res.*, **33**, 7120–7128.

Löytynoja,A. and Goldman,N. (2008) Phylogeny-aware gap placement prevents errors in sequence alignment and evolutionary analysis. *Science*, **320**, 1632–1635.

Lunter,G. *et al.* (2008) Uncertainty in homology inferences: assessing and improving genomic sequence alignment. *Genome Res.*, **18**, 298–309.

Needleman,S.B. and Wunsch,C.D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, **48**, 443–453.

Notredame,C. (2007) Recent evolutions of multiple sequence alignment algorithms. *PLoS Comput. Biol.*, **3**, e123.

Notredame,C. *et al.* (2000) T-coffee: a novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.*, **302**, 205–217.

Nuin,P.A.S. *et al.* (2006) The accuracy of several multiple sequence alignment programs for proteins. *BMC Bioinformatics*, **7**, 471.

Ogdenw,T.H. and Rosenberg,M.S. (2006) Multiple sequence alignment accuracy and phylogenetic inference. *Syst. Biol.*, **55**, 314–328.

Pei,J. and Grishin,N.V. (2001) Al2co: calculation of positional conservation in a protein sequence alignment. *Bioinformatics*, **17**, 700–712.

Roettger,M. *et al.* (2009) A machine-learning approach reveals that alignment properties alone can accurately predict inference of lateral gene transfer from discordant phylogenies. *Mol. Biol. Evol.*, **26**, 1931–1939.

Saitou,N. and Nei,M. (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, **4**, 406–425.

Shan,Y. *et al.* (2003) Automatic recognition of regions of intrinsically poor multiple alignment using machine learning. In *Proceedings of the 2003 IEEE Bioinformatics Conference (CSB2003)*, pp. 482–483.

Sing,T. *et al.* (2005) ROCR: visualizing classifier performance in R. *Bioinformatics*, **21**, 3940–3941.

Smith,T.F. and Waterman,M.S. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.

R Development Core Team (2009) *R: A Language and Environment for Statistical Computing*. Vienna, Austria. ISBN 3-900051-07-0.

Thompson,J.D. *et al*. (1994) Clustalw: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.*, **22**, 4673–4680.

Thompson,J.D. *et al.* (1999) A comprehensive comparison of multiple sequence alignment programs. *Nucleic Acids Res.*, **27**, 2682–2690.

Thompson,J.D. *et al.* (2005) BaliBASE 3.0: latest developments of the multiple sequence alignment benchmark. *Proteins*, **61**, 127–136.

Van Walle,I. *et al.* (2004) Align-m–a new algorithm for multiple alignment of highly divergent sequences. *Bioinformatics*, **20**, 1428–1435.

Wong,K.M. *et al.* (2008) Alignment uncertainty and genomic analysis. *Science*, **319**, 473–476.