Using Cost-Sensitive Learning to Determine Gene Conversions

Mark J. Lawson, Lenwood Heath, Naren Ramakrishnan, Liqing Zhang

Department of Computer Science, Virginia Tech., USA
(malawso4, heath, naren, lqzhang)@vt.edu

Abstract. Gene conversion, a non-reciprocal transfer of genetic information from one sequence to another, is a biological process whose importance in affecting both short-term and long-term evolution cannot be overemphasized. Knowing where gene conversion has occurred gives us important insights into gene duplication and evolution in general. In this paper we present an ensemble-based learning method for predicting gene conversions using two different models of reticulate evolution. Since detecting gene conversion is a rare-class problem, we implement cost-sensitive learning in the form of a generated cost matrix that is used to modify various underlying classifiers. Results show that our method combines the predictive power of different models and is able to predict gene conversion more accurately than any of the two studied models. Our work provides a useful framwork for future improvement of gene conversion predictions through multiple models of gene conversion.

1 Introduction

Gene conversion is a process by which all or part of the sequence of a gene is changed to match the sequence of another gene. Figure 1 gives a schematic demonstration of how gene conversion between two sequences occurs. Its role in both short-term and long-term evolution of genes and genomes cannot be overemphasized. Gene conversion is an important process that can affect the evolutionary fate of duplicated genes. For instance, gene conversion can affect how similar two duplicated genes remain after gene duplication. After gene duplication, depending on which gene is the donor, any mutations in one gene could either be made to disappear or be transferred to the other copy through gene conversion. On the one hand, very frequent gene conversion can lead to two highly identical duplicated genes even long after their duplication. This supports the false impression that the two genes arose from a recent gene duplication as their sequences are highly similar. On the other hand, gene conversion can boost the amount of variation and divergence between duplicated genes, e.g., in the major histocompatibility complex (MHC) genes [1].

Detecting gene conversion is computationally challenging. A number of methods have been proposed for detecting recombinations, of which gene conversion is one type. However, to our knowledge, only a couple of programs (e.g. GENECONV [2]) are specifically designed for detecting gene conversions. Moreover, previous comparisons of the power of existing software to test for recombination have indicated that, while some perform better than others, no single

program is universally superior [3]. Similarly, little is known about which program is the best for gene conversion prediction. To address this problem and to take advantage of the existing programs of recombination prediction, we propose an ensemble based learning method to combine optimally the predictions from different programs to form a unified program prediction and to boost the power of gene conversion prediction.

As in real life, the incidence of gene conversion in a gene family may be restricted to only a few members, and thus the majority cases are gene pairs that do not exhibit gene conversions. Prediction of gene conversion thus falls into the rare-class prediction problem, which is a common phenomenon and has been an issue with many classification problems [4]. Unfortunately, the majority of classifiers will label all data objects as belonging to the majority class. Thus a high accuracy is achieved, however none of the rare-class data objects are correctly identified. There are a variety of ways to deal with rare-class datasets (a paper by Sun et al. provides a nice overview [5]) but for our purposes we went with an implementation of the MetaCost algorithm [6] as our cost-sensitive learner.

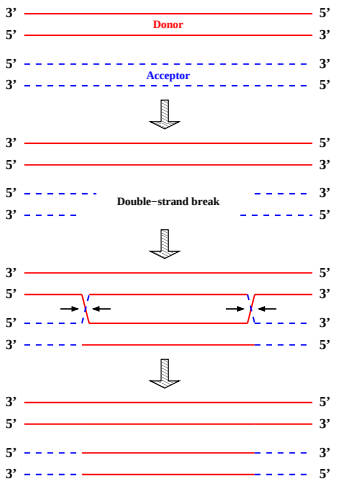

Fig. 1. Gene conversion between two sequences

In this study, we investigated the use of cost-sensitive learning, to combine the output of two programs that are specifically designed for detecting gene conversion, GENECONV [2] and Partimatrix [7]. Our preliminary studies indicate that the cost-sensitive classifier performs better than either of the two programs.

2 Methods

A variety of programs exist that deal with gene conversions and recombination. However few programs exist that provide a definitive prediction on between which two sequences in a set of sequences a gene conversion occurred. Two such programs are GENECONV [2] and Partimatrix [7] which we elaborate next.

GENECONV was developed specifically to identify gene conversions. Given an aligned file of sequences, it will give a prediction of what sequence fragments have the highest, unique similarity between two sequences. It then presents p-values that show the significance of these paired sequence fragments. p-values are determined both globally and pairwise. Global p-values are determined by comparing each fragment with all other possible fragments, thus establishing the overall uniqueness of the paired fragments. Pairwise compares each fragment with the maximum score that would be expected had gene conversion not occurred.

Partimatrix is designed to identify cases of gene conversion and recombination and analyze anomalous phylogenetic history in a multiple sequence align-

ment. For each sequence pair it determines a support score for whether a recombination has occurred and a conflict score that identifies the uniqueness of this recombination. If the support score is significantly higher than the conflict score, this represents evidence for a recombination event between these sequences.

The basic idea behind ensemble-based learning is to use multiple tools that predict, for example, gene conversion, and to use these tools in tandem to achieve prediction results better than each tool could produce alone. Neither GENECONV nor Partimatrix is 100% accurate when it comes to determining gene conversions and preliminary work we did with these programs showed that they would identify correct gene conversions at different times. After running extensive evaluations, we determined that GENECONV performs better in situations where the gene conversion event was recent and the sequences were not very diverged. Partimatrix on the other hand performs better on "ancient" duplication events where the sequences are also more diverged. So through combining them and also features that are measurable that may be associated with gene conversion, we hoped to create an ensemble classifier that would predict gene conversions more accurately than both of these programs.

Each row of attributes used in the classifier is based on data from a pair of sequences. This means that we must divide each set of sequences into a set of sequence pairs. So a set of six sequences for instance would constitute 15 sequence pairs and thus 15 rows of attributes. The attributes include the following data values: average GC content of both sequences, sequence identity between the sequences, global and pairwise p-values from GENECONV, and the support and conflict scores from Partimatrix.

Especially when considering each row of attributes as a pair of sequences, the likelihood of there being a gene conversion is very slim. Due to this fact, we looked into methods of creating a classifier that can handle learning a rare-class. For this we implemented a version of the MetaCost classifier first proposed by Domingos [6]. This method takes a cost matrix and applies it to a base classifier in order to adjust the weights of the training data.

Since our situation is a two-class problem (a gene pair is either labeled as gene conversion or no gene conversion), the cost matrix is a 2x2 matrix (as seen in Table 1). Each of the cells corresponds to the following values: true positives, false negatives, false positives, and true negatives. Within each of these cells, the user can then set the penalty for misclassifiying a row or potentially the reward for correctly classifying one. This cost matrix is then applied to a base classifier.

Table 1. Cost Matrix

True Positive (Correctly Identified Gene Conversions)	**False Positive** (Incorrectly Identified Gene Conversions)
False Negative (Incorrectly Identified No Gene Conversions)	**True Negative** (Correctly Identified No Gene Conversions)

The difficulty with this approach however is selecting the right cost matrix. In our situation, we definitely want to penalize false negatives (situations in which there was a gene conversion, yet the pair was classified as having none) and reward true positives. However, we do not want this to increase the amount

of false positives too much as well, so a penalty must also exist for them. Finding this balance is certainly non-trivial and we have developed a method to do it automatically based on a greedy-search approach.

Starting with a basic, default cost matrix in which the penalty is 1 for both false positives and false negatives and 0 for both true positives and true negatives, we determine how the base classifier does after being trained on the training data and tested on an initial test set. We then increase the penalty of false negatives and false positives by 1 and decrease the reward value for true positives by 1 and every combination of these. We do not adjust the reward for true negatives as this represents the majority class (correctly identified pairs with no gene conversion). After seeing how the classifier performs on every combination of incrementing/decrementing the cost matrix values (7 in total), we then take the best combination and then proceed using that as the base matrix. The determination of best is done through the use of the F-measure (= 2 / (1 / Recall + 1 / Precision)), which represents the harmonic mean between recall and precision. Because accuracy is not a good metric in rare-class predictions, a common approach is to use two different metrics: recall and precision [5, 8, 9]. Recall (=True Positives / (True Positives + False Negatives)) "measures the fraction of positive examples correctly predicted by the classifier", and precision (=True Positives / (True Positives + False Positives)) "determines the fraction of [data objects] that actually turns out to be positive in the group the classifier has declared as a positive class" [10]. These are better metrics than accuracy as they indicate how many of the rare-class are identified while still indicating a balance between true and false positives. However, the disadvantage is that sometime it can be difficult to compare two metrics simulatenously and find a classifier that give the best values for both metrics. We therefore adopted the single metric–F-measure to evaluate different classifiers. The cost matrix that produces the best F-measure is used subsequently as the best cost matrix. Since not every iteration produces a best cost matrix that is better than the previous best cost matrix, we keep track of an overall best cost matrix. This can then be repeated for a user-specified number of iterations (we used 30).

We used four base classifiers as part of our experiments. The first is an implementation of the NaiveBayes learner. The second is J4.8 which is a variation of the C4.5 decision tree learner. The third is PART which is a rule-based learner that creates rules based on partial C4.5 decision trees [11]. The fourth is called JRip which is an implementation of the RIPPER algorithm [12]. We implemented these classifiers in the Weka learning environment [13].

As few examples of actual gene conversion events exist, we had to generate our own training data. The generation of the sequences was done in similar fashion to that by Marais [14]. Basically a set of sequences was generated from a randomly generated phylogenetic tree, thus creating a simulated gene family. These sequences are mutated over time and a gene conversion event is inserted between two sequences. Since genes that exhibit gene conversion are known to have increased GC levels, we included a GC bias when inserting the sequence fragment into the acceptor gene. These sequences were aligned with MUSCLE [15].

We generated two separate training sets under different conditions. The first set (SET1) was done under conditions that mirror typical features of gene conversions [16]. In this case GENECONV performs very well under these conditions (and Partimatrix not so well), we created a second dataset (SET2). Recall, Partimatrix performs better in conditions where the sequences are highly diverged and the gene conversion event happened early (i.e. an ancient gene conversion event). This second dataset was created to reflect this situation. In SET1 the overall average sequence identity was 0.83 (with values ranging between 0.99 and 0.66) and in SET2 the overall average identity was 0.49 (with values ranging between 0.6 and 0.39). Each training dataset contained 500 sets of sequences. Each set of sequences contained 6 sequences, in which one gene conversion has taken place. For each training set, two additional test sets of 100 sequences were generated, one to determine the cost matrix and one to test the cost matrix.

3 Results

Table 2. SET1

Classifier	TP	FP	TN	FN	Accuracy	Recall	Precision	F-measure
Perfect	139	0	2051	0	1	1	1	1
Just Say No	0	0	2051	139	0.937	0	UNDEF	UNDEF
GENECONV Strict	102	4	1448	35	0.975	0.745	0.962	0.840
GENECONV LP	123	57	1395	14	0.955	0.898	0.683	0.776
Partimatrix	9	137	1315	128	0.833	0.066	0.062	0.064
G-or-P	128	191	1261	9	0.874	0.934	0.401	0.561
NaiveBayes	122	58	1394	15	0.954	0.891	0.678	0.770
PART	107	5	1447	30	0.978	0.781	0.955	0.859
J4.8	109	11	1441	28	0.975	0.796	0.908	0.848
JRip	111	9	1443	26	0.978	0.810	0.925	0.864

Table 3. SET2

Classifier	TP	FP	TN	FN	Accuracy	Recall	Precision	F-measure
Perfect	150	0	2250	0	1	1	1	1
Just Say No	0	0	2100	150	0.933	0	UNDEF	UNDEF
GENECONV Strict	1	8	2092	149	0.930	0.007	0.111	0.014
GENECONV LP	5	68	2032	145	0.905	0.033	0.068	0.045
Partimatrix	15	135	1965	135	0.880	0.100	0.100	0.100
G-or-P	19	197	1903	131	0.854	0.127	0.088	0.104
NaiveBayes	8	75	2025	142	0.904	0.053	0.096	0.069
PART	35	214	1886	115	0.854	0.233	0.141	0.175
J4.8	23	160	1940	127	0.872	0.153	0.126	0.138
JRip	40	265	1835	110	0.833	0.267	0.131	0.176

The results for the various classifiers can be seen for SET1 and SET2 in Table 2 and Table 3, respectively. The upper part represents the "basic classifiers" while the lower part represents the classifiers that were used as the base classifiers with the MetaCost implementation and the generated cost matrix. The basic classifiers are as follows: "Perfect" represents an ideal classifier and is

included to see how each of the classifiers compare to it. "Just Say No" represents a classifier where every gene pair is considered to have no gene conversion. As can be seen here, a classifier like this has a relatively high accuracy, but a recall of 0 and an undefined precision (due to a division by zero) and F-measure. "GENECONV Strict" represents using GENECONV with its default settings and only evaluating the global p-values. "GENECONV LP" also uses the local pairwise p-values. Partimatrix does not actually give a finite prediction on whether a gene conversion has occurred so we took the pair that has the lowest conflict score as the pair that exhibits gene conversion. "G-or-P" represents a basic combining of the two classifiers based on one rule: if either GENECONV LP or Partimatrix says a gene conversion has occurred, then the pair is labeled as having a gene conversion. In addition, since Partimatrix does not perform as well on sequences that are too similar it gave no results for some pairs in SET1. These pairs were excluded and a total of 2190 pairs were used instead of 2250.

In SET1, GENECONV performs well. "GENECONV Strict" has a high accuracy, even higher than the "Just Say No approach". However, through our method we are able to increase the amount of true positives, increase the accuracy, and most importantly, increase the F-measure. The best performers are JRip and PART, which is not surprising as they are rule-based classifiers and rule-based classifiers are known to perform well on rare-class data [10]. Both have a higher F-measure than "GENECONV Strict", a higher accuracy, and both identify more true positives. J4.8 does well too and identifies more true positives as PART, but more false positives as well. Of all the cost matrix classifiers, NaiveBayes identifies the most true positives, but is hindered by the amount of false positives it identifies.

In SET2, GENECONV performs poorly. Partimatrix identifies more gene conversions and G-or-P has the best F-measure of these basic classifiers. This set also shows the shortcoming of using accuracy as a metric as the "Just Say No" approach would appear to be the best classifier. Among the cost matrix classifiers, the NaiveBayes classifiers performs quite poorly. It has an F-measure lower than G-or-P, so it shows no improvement over a basic classifier (it does not identify more gene conversions correctly either). But the rule-based classifiers again perform quite well, with both identifying more gene conversions and having higher F-measures than any of the basic classifiers. In fact, aside from NaiveBayes, all classifiers exhibit both a higher recall and a higher precision than the basic classifiers, showing a definite improvement.

Table 4. Cost Matrices

NaiveBayes SET1	-3	2	NaiveBayes SET2	-2	1
	2	0		2	0
PART SET1	-4	5	PART SET2	-3	19
	3	0		1	0
J4.8 SET1	-2	4	J4.8 SET2	0	4
	3	0		1	0
JRip SET1	-4	6	JRip SET2	0	7
	1	0		1	0

Table 4 shows the cost matrices that were determined for each classifier by the greedy-based approach and subsequently used to make gene conversion predictions. It indicates that a cost matrix is highly dependent on both the classifier and the data being used. No classifier has the same cost matrix across both datasets and no dataset has a cost matrix that is best for more than one classifier. In fact, all cost matrices that were determined by our approach are unique.

Many trends are noticeable when looking at the cost matrices. If we exclude NaiveBayes, which performed poorly on both datasets, all classifiers have higher penalties for false negatives than false positives. This makes sense as the initial problem most classifiers have when dealing with rare-class problems is identifying those rare-class members and reducing the number of false negatives.

Further interesting to note are trends in the cost matrices for each classifier when comparing them across the two datasets. The reward for true positives for SET1 is generally "greater" (i.e., the lower the value is, the greater the reward) than that for SET2. This seems to reflect the relative comparison of the true positives between SET1 and SET2 as the number of true positives predicted by GENECONV and Partimatrix in SET1 are much higher than that in SET2 (see Tables 2 and 3). Moreover, that the penalty for false positives in SET2 is higher or equal to that in SET1 seems also to reflect the relative comparison of the number of false positives in SET1 and SET2; the number of false positive predictions in SET2 are slightly higher than that in SET1. Comparatively, the difference in the penalty for false negatives between SET1 and SET2 does not seem to be reflected by the difference in the number of false negative predictions between the two sets.

4 Discussion

In our analysis we have shown how a greedy-search for a best cost matrix can be a quite effective method to deal with a rare-class dataset and help with the identification of gene conversions. This method paired with a rule-based classifier seems to be an effective and robust pairing as in both datasets JRip and PART were the best classifiers (with JRip usually outperforming PART). Through our method we have not only improved upon each of the programs GENECONV and Partimatrix but also on their basic combination (G-or-P). This represents an exciting step forward in the accurate prediction of gene conversions.

An interesting observation is the necessity for a way to generate a best cost matrix. As revealed by our analysis, a cost matrix that maximizes the F-measure is highly dependent on both the dataset and the classifier being used. No cost matrix was the same across datasets or classifiers. The majority of research on cost matrices deals with how to implement them [6] and not with how to generate a cost matrix. However, due to this dependency on both data and classifier, it is certainly not a trivial task and further research into generating an "ideal" cost matrix in an efficient and accurate manner is warranted.

SET1 represents a set of sequences in which the gene conversion occurred relatively recently in the life-cycle of the gene family whereas the sequences in SET2

had a gene conversion that was earlier and thus more obscured by mutations that have occurred in the sequences since. Our method shows more improvement in SET2 than it does in SET1. In SET1 the best improvement is approximately 0.024 better in terms of F-measure (JRip better than GENECONV Strict). But in SET2, we have an F-measure improvement of greater than 0.07 (JRip better than G-or-P). This shows that while our approach does improve prediction in both situations, it improves the identification of "older" gene conversions more than more recent ones. Future studies can be directed to better identify these relative ancient gene conversion by taking account of sequence mutation patterns and also incorporating additional gene conversion prediction models.

References

1. Yeager, M., Hughes, A.L.: Evolution of the mammalian mhc: Natural selection, recombination, and convergent evolution. Immunological Reviews **167** (1999) 45–58
2. Sawyer, S.: Statistical tests for detecting gene conversion. Molecular Biology and Evolution **6**(5) (1989) 526–538
3. Posada, D., Crandall, K.A.: Evaluation of methods for detecting recombination from dna sequences: Computer simulations. Proceedings of the National Academy of Sciences of the United States of America **98**(24) (2001) 13757–13762
4. Chawla, N., Japkowicz, N., Kolcz, A.: Editorial: special issue on learning from imbalanced datasets. SIGKDD Explorations Special Issue on Learning from Imbalanced Datasets (2004)
5. Sun, Y., Kamel, M.S., Wong, A.K.C., Wang, Y.: Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition **40** (2007) 3358–3378
6. Domingos, P.: Metacost: A general method for making classifiers cost-sensitive. Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining (KDD-99) (1999)
7. Jakobsen, I.B., Wilson, S.R., Easteal, S.: The partition matrix: Exploring variable phylogenetic signals along nucleotide sequence alignments. Molecular Biology and Evolution **14**(5) (1997) 474–484
8. Joshi, M.V., Kumar, V., Agarwal, R.C.: Evaluating boosting algorithms to classify rare classes: comparison and improvements. ICDM (2001)
9. Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. Proceedings of the 23rd International Conference on Machine Learning (2006)
10. Tan, P.N., Steinbach, M., Kumar, V.: Introduction To Data Mining. Addison-Wesley (2006)
11. Frank, E., Witten, I.H.: Generating accurate rules sets without global optimization. Fifteenth International Conference on Machine Learning (1998)
12. Cohen, W.W.: Fast effective rule induction. Machine Learning: Proceedings of the Twelfth International Conference (ML95) (1995)
13. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Second edn. Elsevier (2005)
14. Marais, G.: Biased gene conversion: implications for genome and sex evolution. Trends Genet **19**(6) (2003) 330–8
15. Edgar, R.C.: Muscle: A multiple sequence alignment method with reduced time and space complexity. BMC Bioinformatics **5** (2004) 1–19
16. Chen, J.M., Cooper, D.N., Chuzhanova, N., Ferec, C., Patrinos, G.P.: Gene conversion: mechanisms, evolution and human disease (2007) Nature Reviews Genetics.