Application of Machine Learning Algorithms for Binning Metagenomic Data

Vasim Mahamuda[†], Manchon U[†], Khaled Rasheed[†,‡]

[†]Department of Computer Science, The University of Georgia, Athens, GA, 30602, USA
[‡]Institute for Artificial Intelligence , The University of Georgia, Athens, GA, 30602, USA

Abstract – *Machine learning algorithms are extensively being used in the field of bioinformatics. With the invent of new genome sequencing techniques there has been a considerable increase in the amount of data available in the biological databases. In research projects, such as determining the structure and function of biological molecules, the role of machine learning is evident. The machine learning algorithm learns a new concept or function from some past experience. When given new examples, it classifies them based on the concept learned. In this article, our goal is to empirically evaluate how well the standard machine learning algorithms perform in classifying metagenomic data. Our approach involves classifying the sequence reads into respective species by performing a codon analysis on the DNA sequences and extracting features that are representative of the sequence reads. We compared the performance of six well known supervised learners - decision trees, Naïve Bayes, support vector machines, artificial neural networks, Bayesian networks and decision tables . We also analyzed the performance of three different meta-learners namely bagging, boosting and stacking. Experimental results are described which investigate the technique being presented.*

Keywords: *Bioinformatics, binning, ensemble methods, machine learning, metagenomics, supervised learners.*

1 Introduction

1.1 Overview

Metagenomics is the study of organisms that cannot be cultured in the laboratory. Metagenomic DNA sequences can be found in samples directly extracted from natural habitats such as land, sea water etc. Metagenomics facilitates the study of a large population of microbial organisms such as bacteria, archea, and viruses [1]. Microbial organisms can be found everywhere such as land, sea water and even in the gut of human beings. Extracting the genomic data from such environmental samples enables us to study all the micro-organisms ranging from bacteria, archea and microeukaryotes that are involved in regulating the earth's ecological balance. Bacteria, archea and viruses belong to the group known as prokaryotes - i.e. the organisms which lack a nucleus. The sequences which are directly taken from natural habitats such as land, sea water, or the gut of human beings are known as metagenomes and the study of this sequence data is known as metagenomics [2]. The sequencing of metagenomic data allows us to explore and analyze the organisms which cannot be cultured in the laboratory [1].

Machine learning algorithms are being applied to various application domains that are related to the field of bioinformatics. One such field is metagenomics. The role of bioinformatics in the field of metagenomics is: (1) to find genes for detecting novel proteins; (2) to find evolutionary relationships among organisms. It is obvious that these algorithms are being applied to sequence reads because the data set is large enough and finding a method which is capable of handling such huge amounts of data is required. At the same time the need for computational methods arise because of the fact that they automate or speed up the process. Machine learning is nothing but building a model based on certain data or previous knowledge, which is known as the training data. When we have an unknown/unseen example and want to predict which category/class it belongs to, we do so by using the model which has been built on some previous knowledge.

Consider '*n*' examples of the form $\{(x_1,y_1), \dots , (x_n,y_n)\}$. The function $y = f(x)$ varies for different inputs of '*x*'. The x_i values are vectors of the form $\{x_{i1}, x_{i2}, \dots, x_{in}\}$. These vectors are called features or attributes of x_i. These features can be either discrete valued or continuous. Y_i is known as the target attribute. The target attribute, also called label, can take either continuous or discrete values. If the target attribute takes values from a discrete set of classes $\{1, \dots, K\}$, then it is a classification problem. In case of a regression problem the target attribute will take values from the real line (continuous) [3]. Machine learning algorithms can be applied either to the problem of regression or classification. Regression is used for predicting the values of a continuous variable, whereas classification is used when we classify the examples into a number of discrete categories or classes.

When the learning algorithm is given '*n*' training examples of labeled data it learns the concept and outputs a model. When new/unseen examples, called the test set, are presented to the learned model it outputs a prediction of the target attribute value based on the concept learned. This is known as supervised learning. Some of the supervised learners are decision trees, support vector machines, artificial neural networks. On the other hand unsupervised learning, i.e. learning without a target attribute, usually involves similarity-based approaches. The examples are partitioned into different clusters or classes. When a new example is presented it is assigned one of these clusters based on the similarity. One of the well-known unsupervised clustering algorithms is K-means.

1.2 Related work

Metagenomics is an active area of research which deals with the study of the microbial world. It has been the major area of focus of many recent sequencing projects. Using metagenomics researchers were able to determine the functional role of the molecules and also the chemical reactions taking place which aid in the process of symbiosis. The researchers also focused on the phylogenetic classification of these organisms. Phylogenetic classification helped researchers to determine what other kind of species, genera or phyla are present in the metagenomic sample [4]. The sequence data which is extracted by shotgun sequencing or other functional sequencing methods yields sequences which are fragmented in nature. The reads obtained can belong to either different species or they can belong to the same species with different strains. Gene assembly of reads which belong to different species can be easily done, but assembling genes of same species with different strains is quite a challenging task [5].

The sequence data obtained from such sequencing methods highly depends on the environment from which the sample is taken. In other words, there is a high significant correlation between the sequencing data obtained and the origin of the sample [2]. There are a variety of genome assemblers available such as Arachne [6], PCAP [7], Atlas [8]. All the above mentioned genome assemblers work on the same principal concept of assembling genome sequences from shotgun reads. The assembled fragments are then searched for probable genes. The two famous gene-finding techniques are GLIMMER [9], FgenesB_Annotator [10]. Glimmer is a system for finding genes of prokaryotes such as bacteria, archae and viruses. It uses hidden markov models which are among the most popular statistical models. FgenesB_Annotator is a package which does an automatic annotation of bacterial genomes.

Assigning reads to reference genomes or finding the similarity to known species is termed as grouping or classifying the sequences. Researchers in the metagenomic community term it as “binning” [11]. Using this concept of binning, the reads are assigned their taxa and hence can be classified. There are both supervised and unsupervised learning mechanisms to find genes in reads of sequences or to assign them to reference genomes. In the case of supervised learning the reads should be assigned labels or target attribute values of reference genomes.

There are many different methods by which genes are being discovered. One common way is to take the sequence reads and perform a blast against the known genes. Some of the common supervised programs are BLAST [12] and MEGAN [13]. BLAST is a program which enables comparison of a query sequence with a database of known sequences. It then identifies the sequences which match the query sequence based on some statistical similarity measure of matches. Metagenomic data present challenges here, as there is little knowledge about these organisms and the fragments of sequences are thought to be incomplete. So even with BLAST we cannot obtain accurate similarity searches of the query sequence to the sequences present in the databases. MEGAN is another similarity search based tool. It offers both a graphical and statistical analysis of the sequence. The sequence is assigned the taxa based on sequence alignment. MEGAN assigns the new query sequence to the sequence which has the highest similarity score. Similarity based approaches such as BLAST and MEGAN can be of little help in finding novel genes when we consider metagenomic data as there could be no reference homologues present in the databases [2].

Another way to predict genes is to find them based on the structural analysis of the composition of the DNA sequence. This can be achieved by either finding the coding regions or the non-coding regions or just searching for ORFs in the sequence and then using these features to build statistical models which can predict novel genes. When we look at a DNA sequence we cannot find any difference between the metagenomic data and the data that is used in other genomic projects. To us it may seem as the sequence data of metagenomic samples is similar to that of other sequencing data because at the abstract level we know that sequences are nothing but strings of A, T, G and C. But if we take into consideration patterns, the biological structure, functional role and long range sequences, they differ [14].

Many programs have been developed for predicting genes, which make use of statistical models. Orphelia [15] is one such program which uses a two stage machine learning approach to compute the gene probability. The program uses artificial neural networks to find the probable genes in the given sequences. Orphelia has two different programs which work for sequence lengths of 300 bp and 700 bp. MetaGene [16] is another program which is used for sequence lengths of (~700 bp) and is used to predict genes in prokaryotes. MetaGeneAnnotator [17] is the successor of MetaGene, and is available as a web server application. GeneMark [18] is yet another program which uses unsupervised learning and the algorithm used is iterative hidden Markov models. The program is based on heuristic approaches. Even though all the above programs are available as a web server application MetaGene, MetaGeneAnnotator and GeneMark support only sequence reads which are of (~700 bp) in length where as Orphelia supports sequences of length 300 bp and 700 bp. Our approach supports variable length sequence reads.

The approach used in this article is different from the above mentioned programs as we are trying to classify the sequences into a known set of species rather than trying to find genes in a particular sequence. Our main goal is to assist in the process of finding relationships among species as to how similar they are. This is a novel approach to the problem of classification. This is accomplished by first finding the important attributes present in the sequence data. Then, these features are given as input to a machine learning algorithm. The algorithm learns from the data and presents a model to the user. The model is then validated for correctness, by giving it a set of examples known as the test set, to see how well the model performs on unseen examples. In [11], the authors used Naïve Bayes classifier for assigning reads to respective phylogenetic groups. In our study we have not limited ourselves to just one classifier but tried a variety of different classifiers, both supervised and ensemble learners to determine which classifier performs well on such metagenomic data.

The rest of the article is organized as follows: in Section 2 we introduce our methodology and show how the features are extracted from the sequence data. In Section 3 we present the performance of various algorithms. In Section 4 we discuss future work and the article is concluded in Section 5.

2 Approach

2.1 Data selection

The data which is used in our experiments is available on Comprehensive Microbial Resource [19]. The data is selected in such a way that each species selected has a different genus. We randomly select 75 different species based on the criterion that each species should belong to a different genus. This set of 75 species is equally divided into 3 groups of 25 species each. From each of these groups we remove 10 species to form the set of 15 species. Two sets of experiments are performed for each group: one with the initial set of 25 species and the other with the set of 15 species. As the species selection is stochastic, in order to validate our approach we report the average accuracy of the 3 groups. As shown in Fig. 1, species which are selected for our experiments belong to different genus.

2.2 Data pre-processing

The FASTA file of each species contains a number of different reads of DNA sequences. The number of reads and the average length of each read vary according to the species being selected. The average number of reads of DNA sequences in a file of each species would be approximately 3500 and the length of DNA sequence would be between 300-1000 bp.

The next step in our approach is to extract the features or attributes, for the machine learning algorithms to learn the concept and then to bin them into groups. The attributes or features we selected are the GC content, number of ORFs, uni-base frequency, di-base frequency, average length of ORF. GC content is the most important feature to consider because of the relative stability of the bond between the G and C bases rather than A and T bases [20]. If the DNA sequence has a pattern such that it begins with a start codon (ATG,CTG,GTG, or TTG), which is followed by at least 54 bp, and then ends with a stop codon (TGA,TAG, or TAA) then such a pattern is known as an ‘Open Reading Frame or ORF [15]. Codons in biology are subsequent, non-overlapping triplets. We considered ORFs with overlap as the relative lengths of the sequences are short. By considering ORFs with overlap we get a good count of the number of ORFs in a sequence even though the sequences are short [15].

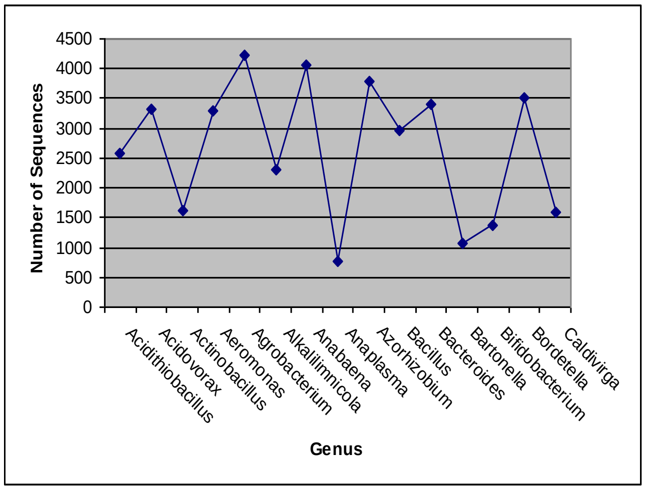
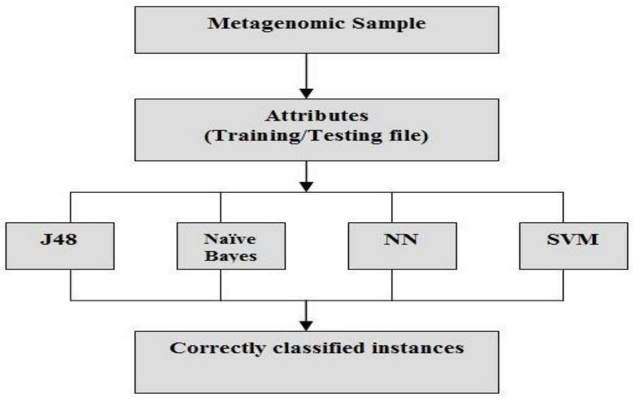


Fig. 1. Shows how sequence lengths are distributed among the different genera.

The DNA sequences which are in the FASTA format have a sequence identifier followed by the sequence itself. We first compute the GC content and then find the probable ORFs for each read of DNA sequence. As the sequence can be translated in six different ways, we find the ORFs in all possible frames (3 on positive strand, 3 on the minus strand). Now iterating through each ORF we look for codons with a pattern. Uni-base frequency is the term, which we use for codons which contain only one unique nucleotide, namely AAA, TTT, GGG, CCC. Di-base frequency is the frequency of the codons which contain two unique nucleotides, such as ATA, TTA, GGC, CGC and so on. We did not consider the feature tri-base frequency (codons which differ by all three nucleotides) as the features uni-base, di-base and tri-base are complementary.

This set of five features along with the species label (i.e. to which species the sequence belongs) forms the input to the machine learning algorithm. One sample tuple would be of the form (number of ORFs, uni-base frequency, di-base frequency, average length of ORFs, GC content, name of the species). Here, the name of the species is the target attribute which is discrete valued. In this way we process all the sequences such that for each sequence we have a tuple which gives a statistical representation of the underlying data. Fig. 2 shows a diagrammatic representation.



Metagenomic Sample
Attributes (Training/Testing file)
J48
Naïve Bayes
NN
SVM
Correctly classified instances

Fig. 2. Step-by-step illustration of our approach.

After the data has been processed and the features extracted the next step is to present it as an input to some machine learning algorithm.

2.3 Learning methods

The machine learning algorithms which we use are decision trees, decision tables, neural networks, support vector machines, and Naïve Bayes, Bayesian networks. Ensemble learners such as bagging, boosting and stacking are also used. We use the University of Waikato's WEKA [21] software which is an open source data mining software written in Java, to run the experiments. The Weka package has implementations of various popular machine learning algorithms.

We use a validation method approach wherein the data is randomly split into two sets of 80% and 20% known as the training/validation set and the testing set. The algorithm builds the model based on the training set and then the test set is presented to the model to see how well it performs. Another way of validating the data is by using N - fold cross validation approach. In this approach the data is partitioned into N disjoint subsets, trained on N-1 subsets and the remaining one set is used for validation. This process is repeated for N times and the average accuracy is reported. Cross validation is generally used when the size of the data set is small which is not the case in this research.

The first classifier used is the decision tree. The target attribute is the name of the species to which the sequence belongs. The target attribute in this case is discrete valued, and hence decision trees can be used. The algorithm used is J48 which is an extension to the C4.5 algorithm, developed by Quinlan in 1993. The construction of the tree follows a top down approach. All the attributes are evaluated by a statistical measure called information gain [22]. The attribute with the highest value of information gain will be the root of the tree. The algorithm is then called recursively for every sub-tree. To avoid overfitting of the data the tree is pruned. Overfitting is a phenomenon in which the model is able to predict correct outputs for the training examples but does not generalize well to unseen examples. This usually happens when training is performed for a long time or when the training examples are very rare. It fits the model very well to the training data and hence the performance on the training set increases considerably but it undermines the performance on new unseen data. Similar to decision trees, decision tables also involve selecting the attributes by discarding the irrelevant attributes from the decision table.

The second classifier we used to evaluate the data is Naïve Bayes. It is a supervised learning algorithm in which the target function can take in any discrete value. It uses a probabilistic model where the probabilities of the attributes are calculated. When given a new example to classify the classifier assigns the most probable target attribute value given the other (non-target) attributes [22]. Bayesian network is a probabilistic graphical model that represents a set of random variables and their conditional independencies via a directed acyclic graph. Bayesian networks are similar to naïve Bayes except for the fact that they make weaker assumptions for conditional independance between attributes. Both Naive Bayes and Bayesian networks are relatively fast and take just few minutes to train and classify the instances.

The third classifier used is the multilayered perceptron Artificial Neural Network (ANN) trained by the back-propagation algorithm. ANN's are useful when the training data has lot of noise. The squared error between the target output and the network values is minimized by adjusting the weights [22]. It takes 10 times as long to train a neural network (on the order of few hours on a desktop) as compared to Naïve Bayes which in turn is slower than decision trees. To prevent overfitting of data we changed the default values of the parameters. We used a validation set size of 25%, validation threshold of 20 epochs and trained the network for 500 epochs. This will allow the network to stop on the validation error and not because the number of epochs have been exhausted. We chose these parameters because the domain exhibits a 'saw tooth' behavior. The threshold needs to be kept high and the number of epochs needs to be increased. It may help in such a domain to increase both the learning rate and the momentum term to speed up the training process.

Support Vector Machines (SVMs) are really popular in application domains where the data is linearly separable. It can also be applied to problems of either regression or classification. We used the Sequential Minimal Optimization algorithm which has been implemented in Weka. Training times are faster as compared to ANN but is slow compared to decision trees and Naïve Bayes.

Ensemble methods aim at enhancing the performance of a given statistical model. Ensemble learning is to combine models of either the same type or of a different type. ‘Bagging’, ‘boosting’ and ‘stacking’ all fall into the category of ensemble learners or meta-learners. Bagging and boosting combine models of the same type (usually called the base-classifier type) while stacking combines models of different types (i.e. generated by different algorithms). Bagging is also known as boot strap aggregation. For bagging and boosting we used J48 as the base-classifier because it gave the best performance in the first set of experiments. For stacking we combined a total of five classifiers. The base-classifiers (level-0) used for stacking are Naïve Bayes, SVM, Bayesian network, decision table, and the level-1 or meta-level classifier was J48. There is no hard and fast rule to select which classifiers are to be used for level-0, level-1 classifiers so we selected the faster classifiers because stacking requires much more time than single-classifier learning.

3 Experiments & Results

3.1 Evaluation approach

The performance of our classifiers is tested using validation. We used an 80-20 split of the data, where 80% of the data is used for training/validation and 20% of the data as the test set [23]. The classifier is evaluated on how well it classifies the 20% of the examples based on the learned concept. Other metrics based on which a classifier can be evaluated include Sensitivity, Specificity, and Accuracy. True positives (TP) are the number of positive examples classified as positive. False negatives (FN) are the number of positive examples which are classified as negatives. True negatives (TN) are the number of negative examples which are classified as negative. False positives (FP) are the number of negative examples which are classified as positive [22].

We can define sensitivity and specificity as statistical measures of performance of the binary classification test, namely:

$$Sensitivity = TP/(TP + FN) \quad (1)$$

$$Specificity = TN/(TN + FP) \quad (2)$$

$$Accuracy = (TP + TN)/(TP + FN + TN + FP) \quad (3)$$

We evaluate our results based on the metric of accuracy, which can be thought of as the proportion of instances which are correctly classified.

3.2 Results

Experiments are performed for 3 groups of 15 and 25 species respectively. The number of instances given to the classifiers is approximately 45,000 in the case of 15 species and approximately 77,000 for 25 species. Among meta-learners bagging and boosting with base-classifier as J48 perform better than stacking. Also, training time of J48 is fast as compared to all the other learning algorithms, the worst being ANN which takes hours to build the model.

A closer look at the resulting decision tree reveals that the attribute GC content is always selected as the root of the tree. As GC content is an important feature in determining the functional roles of the species, it has the highest information gain and hence is selected as the root of the decision tree.

Table I: Accuracy of supervised learners on three independent groups of 15 species

Algorithm	1st Group	2nd Group	3rd Group	Average
J48	92.7218	90.6375	92.3518	**91.9037**
Bayes Net	91.2564	91.0585	92.2176	**91.5108**
Decision Table	92.0281	89.3444	91.0466	**90.8036**
ANN	87.4951	83.1997	85.4111	**85.3686**
NB	81.8875	84.0517	86.4967	**84.1453**
SVM	82.6886	74.4611	83.6545	**80.2680**

Table II: Accuracy of meta-learners on three independent groups of 15 species.

Method	1st Group	2nd Group	3rd Group	Average
Bagging	93.0148	91.4395	93.0471	**92.5004**
Boosting	92.8292	91.5297	93.1447	**92.5012**
Stacking	92.1551	89.9559	91.4979	**91.2029**

Table III: Accuracy of supervised learners on three independent groups of 25 species.

Algorithm	1st Group	2nd Group	3rd Group	Average
Bayes Net	91.1732	86.0028	86.4445	**87.8735**
Decision Table	91.0554	84.5153	85.7273	**87.0993**
J48	88.2888	86.1439	86.4586	**86.9637**
ANN	84.9694	76.8402	81.9026	**81.2374**
NB	78.5845	75.2437	77.2411	**77.0231**
SVM	76.489	63.5804	69.8024	**69.9572**

Table IV: Accuracy of meta-learners on three independent groups of 25 species.

Method	1st Group	2nd Group	3rd Group	Average
Bagging	92.4611	87.2852	87.7522	**89.1661**
Boosting	92.3747	87.5417	87.0491	**88.9885**
Stacking	90.6	84.791	85.6641	**87.0183**

As we can see from Table I., decision trees gave the best performance for 15 species. For 25 species, Table III shows that the Bayesian network performs best but the performance of decision trees is still comparable. Also, when we compared the performance for 15 and 25 species, we expected the accuracy with 15 species to be much better compared with 25 species. This is because in a classification problem with 15 species the random guessing accuracy is expected to be 1/15 or 6.66%, which is higher than that for 25 species (only 4%). However, we noticed that the accuracy for 25 species was not much worse than that of 15 species.

For meta-learners, bagging and boosting perform better than stacking. From the results we can see that the performance of bagging and boosting is higher than J48. We also tried bagging and boosting other classifiers such at the Bayesian network but the best results were obtained with J48 as the base classifier. Also, we can see that the performance of stacking is almost equal to that of J48. From Tables II and IV, we can see that the accuracy of the meta-learners for 15 species is higher than that of 25 species.

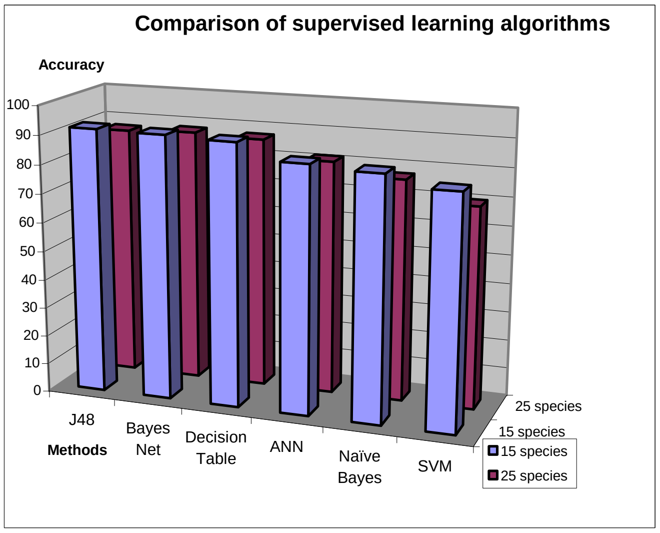


Fig 3. Comparison of accuracies of 15 and 25 species for supervised learners.

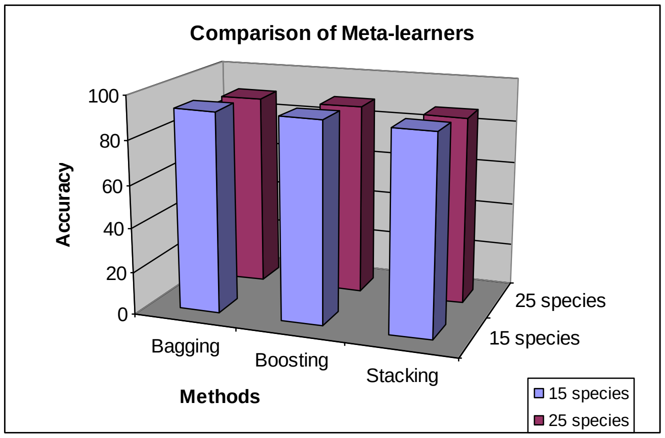


Fig. 4 Comparison of accuracies of 15 and 25 species for meta-learners.

In [11], the authors used the Naïve Bayes classifier for assigning reads to phylogenetic groups. The features used are k-mer frequencies, and they vary ‘k’ from 7 to 10. They introduced prior knowledge of genomes by a preliminary 16S survey. The Naïve Bayes classifier without any prior knowledge performs with an accuracy of 32.9% for 7-mers and it gradually increases to 43.9% as number of k increases from 7 to 10. To improve the performance they used prior knowledge and the assignment accuracies range from 57.4% to 60.5% as the number of k-mers increase from 7 to 10. For 15 species we have an accuracy of 83.35% using Naïve Bayes, with only five features and without the use of any prior knowledge.

4 Future work

An extension to the work would be to implement a two phase approach of classifying the sequence reads into respective species first and then by using some statistical analysis determine whether that sequence contains a gene or not. We also plan to evaluate our results by adding more features. Feature selection approaches such as wrapper methods can then be used to improve the performance of the various machine learning algorithms.

Finally, we are currently working on applying our approach to sequence reads which are unknown. We take all the sequence reads present in 15 species and to that set of sequence reads we add reads of sequences from species which are selected randomly and which are distinct from the 15 species that are selected above. We check to see how many of these unknown sequence reads actually get classified as unknown. This way we can simulate a real world problem of mixed sequence reads. We can then evaluate the scalability of the methods and see how well the algorithms perform on such mixed data.

5 Conclusion

The motivation behind this work is to see which machine learning algorithm does well on classifying or binning the metagenomic data. We address an important problem in the field of bioinformatics, which is to observe relationships among species. Our approach is to apply well known supervised learners and also meta-learners to determine which of the evaluated learning approaches is most accurate on this task. The work is significant as we always have samples of sequences which are mixed or which belong to different species with metagenomic data. When we want to separate the sequences so that they belong to different phylogenetic groups, then this work can help significantly to decide which machine learning algorithms to use.

In this article, we presented a novel approach for classifying the sequences into the respective species by well known machine learning algorithms. The features selected, though only few, were clearly very good at differentiating the data. The accuracy for all the three groups of each number of species (15 or 25) is close, indicating the consistent performance of the learning algorithms. Our results suggest that decision trees can be used to bin the sequence reads in a metagenomic sample. Furthermore, the performance of decision trees can be slightly improved by bagging and boosting. Also, the results indicate that machine learning algorithms such as decision trees, Bayesian networks, decision tables, artificial neural networks, support vector machines, Na ïve Bayes can greatly aid in the important task of binning metagenomic data.

6 References

[1] J. Venter, *et al.*, "Environmental genome shotgun sequencing of the Sargasso Sea," *Science*, vol. 304, p. 66, 2004.

[2] J. Wooley, *et al.*, "A Primer on Metagenomics," *PLoS Computational Biology*, vol. 6, 2010.

[3] T. Dietterich, "Ensemble methods in machine learning," *Multiple classifier systems*, pp. 1-15, 2000.

[4] J. Handelsman, "Metagenomics: application of genomics to uncultured microorganisms," *Microbiology and Molecular Biology Reviews*, vol. 68, p. 669, 2004.

[5] V. Markowitz, *et al.*, "An experimental metagenome data management and analysis system," *Bioinformatics*, vol. 22, p. e359, 2006.

[6] S. Batzoglou, *et al.*, "ARACHNE: a whole-genome shotgun assembler," *Genome Research*, vol. 12, p. 177, 2002.

[7] X. Huang, *et al.*, "PCAP: a whole-genome assembly program," *Genome Research*, vol. 13, p. 2164, 2003.

[8] P. Havlak, *et al.*, "The Atlas genome assembly system," *Genome Research*, vol. 14, p. 721, 2004.

[9] A. Delcher, *et al.*, "Improved microbial gene identification with GLIMMER," *Nucleic Acids Research*, vol. 27, p. 4636, 1999.

[10] G. Tyson, *et al.*, "Community structure and metabolism through reconstruction of microbial genomes from the environment," *Nature*, vol. 428, pp. 37-43, 2004.

[11] L. Kuan-Liang, Tsu-Tsung, Wong, Gary Xie, Nicholas W H, "Improving Na ïve Bayesian Classifier for Metagenomics reads assignment," *Biocomp*, pp. 259-264, 2009.

[12] S. McGinnis and T. Madden, "BLAST: at the core of a powerful and diverse set of sequence analysis tools," *Nucleic Acids Research*, vol. 32, p. W20, 2004.

[13] D. Huson, *et al.*, "MEGAN analysis of metagenomic data," *Genome Research*, vol. 17, p. 377, 2007.

[14] J. Wooley and Y. Ye, "Metagenomics: Facts and Artifacts, and Computational Challenges," *Journal of Computer Science and Technology*, vol. 25, pp. 71-81, 2009.

[15] K. Hoff, *et al.*, "Orphelia: predicting genes in metagenomic sequencing reads," *Nucleic Acids Research, vol. 37,* 2009.

[16] H. Noguchi, *et al.*, "MetaGene: prokaryotic gene finding from environmental genome shotgun sequences," *Nucleic Acids Research, vol. 34(19), pp. 5623 - 5630,* 2006.

[17] H. Noguchi, *et al.*, "MetaGeneAnnotator: detecting species-specific patterns of ribosomal binding site for precise gene prediction in anonymous prokaryotic and phage genomes," *DNA research*, vol. 15, p. 387, 2008.

[18] A. Lukashin and M. Borodovsky, "GeneMark. hmm: new solutions for gene finding," *Nucleic Acids Research*, vol. 26, p. 1107, 1998.

[19] *Comprehensive Microbial Resource*. [Online]. Available: http://cmr.jcvi.org/cgi-bin/CMR/shared/Menu.cgi?menu=downloads. [Accessed: April 11, 2010].

[20] J. Raes, *et al.*, "Get the most out of your metagenome: computational analysis of environmental sequence data," *Current opinion in microbiology*, vol. 10, pp. 490-498, 2007.

[21] *WEKA 3- Data Mining with open source Machine Learning software in Java.* [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/. [Accessed: April 11, 2010].

[22] T. M. Mitchell, *Machine Learning*: McGraw-Hill, 1997.

[23] P. Crowther and R. Cox, "A Method for Optimal Division of Data Sets for Use in Neural Networks," *Springer Berlin/Heidelberg*, pp. 1-7, 2005.