A peer-reviewed version of this preprint was published in PeerJ on 24 January 2019.

Eppenhof EJJ, Peña-Castillo L. 2019. Prioritizing bona fide bacterial small RNAs with machine learning classifiers. PeerJ 7:e6304
https://doi.org/10.7717/peerj.6304

Prioritizing bona fide bacterial small RNAs with machine learning classifiers

Erik JJ Eppenhof [1] , **Lourdes Peña-Castillo** Corresp. [2, 3]

[1] Department of Artificial Intelligence, Radboud University Nijmegen, Nijmegen, Netherlands
[2] Department of Biology, Memorial University of Newfoundland, St. John's, Canada
[3] Department of Computer Science, Memorial University of Newfoundland, St. John's, Canada

Corresponding Author: Lourdes Peña-Castillo
Email address: lourdes@mun.ca

Bacterial small non-coding RNAs (sRNAs) are involved in the control of several cellular processes. Hundreds of putative sRNAs have been identified in many bacterial species through RNA sequencing. The existence of putative sRNAs is usually validated by Northern blot analysis. However, the large amount of novel putative sRNAs reported in the literature makes it impractical to validate in the wet lab each of them. In this work, we applied five machine learning approaches to construct twenty models to discriminate bona fide sRNAs from random genomic sequences in five bacterial species. Sequences were represented using seven features including free energy of their predicted secondary structure, their distances to the closest predicted promoter site and Rho-independent terminator, and their distance to the closest open reading frames (ORFs). To automatically calculate these features, we developed an sRNA Characterization Pipeline (sRNACharP). All sevens features used in the classification task contributed positively to the performance of the predictive models. The five best performing models obtained a median precision of 100% at 10% recall and of 60% at 40% recall across all five bacterial species. Our results suggest that even though there is limited sRNA sequence conservation across different bacterial species, there are intrinsic features of sRNAs that are conserved across taxa. We show that these features are exploited by machine learning approaches to learn a species-independent model to prioritize bona fide bacterial sRNAs.

Prioritizing bona fide bacterial small RNAs with machine learning classifiers

Erik JJ Eppenhof[1] **and Lourdes Peña-Castillo**[2]

[1]**Department of Artificial Intelligence, Radboud University, Nijmegen, GE, The Netherlands**
[2]**Department of Computer Science and Department of Biology, Memorial University of Newfoundland, St. John's, NL, Canada**

Corresponding author:
Lourdes Peña-Castillo[2]

Email address: lourdes@mun.ca

ABSTRACT

Bacterial small non-coding RNAs (sRNAs) are involved in the control of several cellular processes. Hundreds of putative sRNAs have been identified in many bacterial species through RNA sequencing. The existence of putative sRNAs is usually validated by Northern blot analysis. However, the large amount of novel putative sRNAs reported in the literature makes it impractical to validate in the wet lab each of them. In this work, we applied five machine learning approaches to construct twenty models to discriminate bona fide sRNAs from random genomic sequences in five bacterial species. Sequences were represented using seven features including free energy of their predicted secondary structure, their distances to the closest predicted promoter site and Rho-independent terminator, and their distance to the closest open reading frames (ORFs). To automatically calculate these features, we developed an sRNA Characterization Pipeline (sRNACharP). All sevens features used in the classification task contributed positively to the performance of the predictive models. The five best performing models obtained a median precision of 100% at 10% recall and of 60% at 40% recall across all five bacterial species. Our results suggest that even though there is limited sRNA sequence conservation across different bacterial species, there are intrinsic features of sRNAs that are conserved across taxa. We show that these features are exploited by machine learning approaches to learn a species-independent model to prioritize bona fide bacterial sRNAs.

INTRODUCTION

Bacterial small non-coding RNAs (sRNAs) are regulatory RNAs (usually between 50 to 250 nucleotides) that are known to play a role in the control of several cellular processes (Storz et al., 2011; Michaux et al., 2014). A multitude of putative sRNAs has been identified in many bacterial species through RNA sequencing (e.g., Grüll et al. (2017); Thomason et al. (2015); Zeng and Sundin (2014); McClure et al. (2014)). The existence of putative sRNAs is usually validated by Northern blot analysis. However, the large amount of novel putative sRNAs reported in the literature makes it impractical to validate each of them in the wet lab. To optimize resources, one would like to first investigate those putative sRNAs which are more likely to be bona fide sRNAs. To do that, we need to computationally prioritize sRNAs based on their likelihood of being bona fide sRNAs. As the inter-species sequence conservation of sRNAs is very limited and most sRNAs are species-specific (Gómez-Lozano et al., 2015; Grüll et al., 2017), sRNA prioritization based on sequence similarity to known sRNAs has a low recall rate. However, predictive models generated by machine learning approaches may be able to detect intrinsic features of sRNA sequences common to a number of bacterial species.

We comparatively assessed the performance of five machine learning approaches for quantifying the probability of a genomic sequence encoding a bona fide sRNA. The machine learning approaches applied were: logistic regression (LR), multilayer perceptron (MP), random forest (RF), adaptive boosting (AB) and gradient boosting (GB). We used data from five bacterial species including representatives from the phyla *Firmicutes* (*Streptococcus pyogenes*), *Actinobacteria* (*Mycobacterium tuberculosis*), and *Proteobac-*

teria (*Escherichia coli*, *Salmonella enterica*, and *Rhodobacter capsulatus*). As input to the machine learning approaches, we provided a vector of seven features per sequence. These features are: the free energy of the predicted secondary structure, distance to their closest predicted promoter site, distance to their closest predicted Rho-independent terminator, distances to their two closest open reading frames (ORFs), and whether or not the sRNA is transcribed on the same strand as their two closest ORFs. Obtaining these sRNA features requires the use of numerous different bioinformatics tools which may be challenging for the average user. To facilitate sRNA characterization, we have developed sRNACharP (**sRNA Char**acterization **P**ipeline), a pipeline to automatically compute these seven features (available at `https://github.com/BioinformaticsLabAtMUN/sRNACharP`). Results from our comparative assessment indicate that it is possible to create a highly accurate and general (i.e., species-independent) model for prioritizing bona fide bacterial sRNAs. To enable other researchers to use one of the best species-independent sRNA predictive models we evaluated, we introduce sRNARanking, a freely available species-independent predictive model aimed at computationally prioritizing putative sRNAs based on their likelihood to be bona fide sRNAs (`https://github.com/BioinformaticsLabAtMUN/sRNARanking`). We expect that together these two tools (sRNACharP and sRNARanking) will facilitate and accelerate the characterization and prioritization of putative sRNAs helping researchers in the field of RNA-based regulation in bacteria to focus in the putative sRNAs most likely to be bona fide sRNAs.

METHODS

Data sets

Published positive instances of bona fide sRNAs were collected for *R. capsulatus* (Grüll et al., 2017), *S. pyogenes* (Le Rhun et al., 2016), and *S. enterica* (Kröger et al., 2012). *S. pyogenes* and *S. enterica* positive instances have all been verified by Northern blot analysis; while, *R. capsulatus* positive instances included, in addition to four experimentally verified sRNAs, 41 homologous sRNAs (i.e., sRNAs that have high sequence similarity to known sRNAs in other bacterial species or were found to be conserved in the genome of at least two other bacterial species). We randomly selected 80% of the positive instances for training, while setting aside the other 20% for validating the models. Ten random genomic sequences (negative instances) were generated using shuffleBed (Quinlan and Hall, 2010) for each of the positive instances. These negative instances were of the same length as the positive instances. We then randomly selected *n* random sequences for training, where *n* is three times the number of positive instances in the corresponding training set. All remaining random sequences were used for validating the models.

Additionally, we collected *E. coli* sRNAs, supported by literature with experimental evidence from RegulonDB (release 9.3) (Gama-Castro et al., 2016), and *M. tuberculosis* sRNAs verified by Northern blot analysis from Miotto et al. (2012). We generated negative instances for these two species as previously mentioned. *E. coli* and *M. tuberculosis* data was used exclusively for validating the predictive models. The number of positive and negative instances per bacterial species used for training and validating the machine learning models is shown in Table 1. Data sets are provided in Additional File 1.

Table 1. The number of positive (bona-fide sRNAs) and negative (random genomic sequences) instances in the data sets used for training and validating the classification models. The NCBI accession number of the genome sequence used is indicated in the first column between brackets. The "Combined" data is made by putting together the training data of *S. enterica*, *S. pyogenes* and *R. capsulatus*.

	Training		Validation	
	Positive Instances	**Negative Instances**	**Positive Instances**	**Negative Instances**
R. capsulatus (NC_014034.1)	36	108	9	342
S. pyogenes (NC_002737.2)	37	110	9	350
S. enterica (NC_016810.1)	90	271	23	859
Combined	163	489	N/A	N/A
E. coli (NC_000913.3)	N/A	N/A	125	1250
M. tuberculosis (NC_000962.3)	N/A	N/A	19	190

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

sRNA Characterization

Each sRNA is represented as a vector of seven numerical features or attributes, as in Grüll et al. (2017). These attributes are:

1. free energy of the sRNA predicted secondary structure,
2. distance to the -10 predicted promoter site in the range of [-150, length of the sequence] nucleotides (nts) (if no promoter site is predicted in that range a value of -1000 is used),
3. distance to the closest predicted rho-independent terminator in the range of [0,1000] nts (if no terminator is predicted within this distance range a value of 1000 is used),
4. distance to the closest left ORF, which is in the range of (-∞, 0] nts,
5. a Boolean value (0 or 1) indicating whether the sRNA is transcribed on the same strand as its left ORF,
6. distance to the closest right ORF, which is in the range of [0, +∞), and
7. a Boolean value indicating whether the sRNA is transcribed on the same strand as its right ORF.

A "left" ORF is an annotated ORF located at the 5' end of a genomic sequence on the forward strand or located at the 3' end of a genomic sequence on the reverse strand (Fig.1). A "right" ORF is an annotated ORF located at the 3' end of a genomic sequence on the forward strand or located at the 5' end of a genomic sequence on the reverse strand.

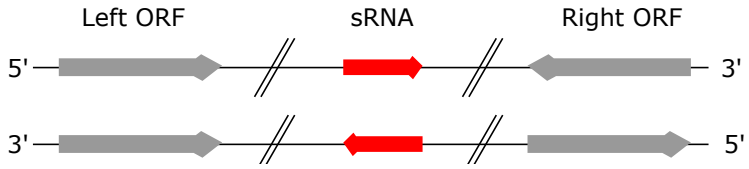

Figure 1. Left and right ORFs. Left ORFs are located at the 5' end of a sRNA on the forward strand or at the 3' end of a sRNA on the reverse strand. Right ORFs are located at the 3' end of a sRNA on the forward strand or at the 5' end of a sRNA on the reverse strand.

To automatically calculate these seven features for a set of sRNAs from a given bacterial species, we developed sRNACharP. As input, sRNACharP requires only a BED file (UCSC website, 2018) with the genomic coordinates of the sRNAs, a FASTA file with the corresponding genome sequence, and a BED file with the genomic coordinates of the annotated protein coding genes (ORFs). sRNACharP is implemented in Nextflow (Di Tommaso et al., 2017) and available at `github.com/BioinformaticsLabAtMUN/sRNACharP`. To ensure reproducible results and reduce installation requirements to the minimum, sRNACharP is distributed with a Docker container (Di Tommaso et al., 2015). sRNACharP uses the following bioinformatics tools (the versions listed within brackets are the ones installed in the Docker container). CentroidFold (Hamada et al., 2009) (version 0.0.15) with parameters `-e ''CONTRAfold''` and `-g 4` is used to predict the secondary structure of the sequences given. Bedtools' slopBed and fastaFromBed (Quinlan and Hall, 2010) (version 2.26) are used to extract the sRNA sequences, and the sequences including 150 nts upstream of the 5' end of the sRNAs in FASTA format. Promoter sites on the sequences including 150 nts upstream of the 5' end of the sRNAs are predicted using BPROM (Solovyev and Salamov, 2011) with default values. Rho-independent terminators are predicted using TransTermHP (Kingsford et al., 2007) (version 2.09) with default values. Alternatively, sRNACharP can take as input, files from the TransTermHP website (`http://transterm.cbcb.umd.edu/cgi-bin/transterm/predictions.pl`). For this study, we downloaded the predicted rho-independent terminators for *S. pyogenes* and *M. tuberculosis* from the TransTermHP website on March 2017. The distances to the closest terminator and the closest ORFs are obtained using bedtools' closest. Finally, R (version 3.4.4) is used to generate the features table.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

Machine Learning Approaches

We assessed the performance of logistic regression (Cox, 1958; Walker and Duncan, 1967), multilayer perceptron (Bishop, 1995; Fahlman, 1988), random forest (Ho, 1995; Dietterich, 2000a; Breiman, 2001) and boosting models (Schapire, 1990) for the task of quantifying the probability of a genomic sequence encoding a bona fide sRNA. Random forest and boosting classifiers are both examples of ensemble learning algorithms (Dietterich, 2000b). The core of the boosting methods lies in iteratively combining outputs of so-called "weak learners", converging to an overall strong learner. Logistic regression (LR) was used in Grüll et al. (2017) and showed to outperform linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA) for this task. We decided to use LR as a baseline to compare the performance of the other classifiers. We chose to compare the other four machine learning approaches (classifiers) because they have shown to perform well on small data sets and they are generally robust to noise (Liaw and Wiener, 2002; Kerlirzin and Vallet, 1993; Ridgeway, 1999).

All the machine learning classification approaches were implemented in the Python programming language version 3.6. Scikit-learn (version 0.19.1) (Pedregosa et al., 2011) was used for the implementation of the logistic regression, boosting and random forest classifiers. The multilayer perceptron classifier was implemented following the pseudoalgorithms provided by Bishop (1995). All the Python scripts were executed on a MacBook Air 2Ghz Intel Core i7 with 8GB of RAM and OS X (version 10.9.5). For each classifier, the "best" parameters were obtained by optimizing the area under the ROC curve (AUC) when performing leave-one-out cross-validation (LOO CV) on the training data.

Logistic Regression

Logistic Regression (LR) learns the parameters β of the logistic function,

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\ldots+\beta_n X_n}}{1+e^{\beta_0+\beta_1 X_1+\ldots+\beta_n X_n}},$$

where $p(X)$ is the probability of an sRNA with feature vector X of being a bona fide sRNA, e is the base of the natural logarithm, n is the number of features, and X_i is the value of feature i. To fit the model, usually the maximum likelihood approach is used. We used the "balanced" mode that automatically adjust class weights inversely proportional to class frequencies in the input data. All other parameters were left to their default values.

Multilayer Perceptron

Multilayer Perceptrons (MPs) are fully connected feed-forward neural networks, with one or more layers of hidden nodes between the input and output nodes (Bishop, 1995; Fahlman, 1988). Except for the input node(s), each node is a neurone with a nonlinear activation function. Each neurone combines weighted inputs by computing their sum to determine its output based on a certain threshold value and the activation function. The output y of the system can be described as

$$y = f(\textstyle\sum_{i=0}^{N} w_i x_i),$$

where $x_1,\ldots,x_N$ represent the input signals, $w_1,\ldots,w_N$ are the synaptic weights and f is the activation function. MPs learn through an iterative process of changing connection weights after processing each part of the data. The most common learning algorithm used for this process is backpropagation (Fahlman, 1988).

The activation function that lead to the largest AUCs on the training data was the logistic sigmoid function. We used the standard backpropagation algorithm with an initial random generation of weights ([-1,1]). As using multiple hidden layers decreased the performance, we decided to use only one hidden layer. The number of hidden nodes explored was in the range from 1 (in that case the model behaves the same as logistic regression) to 1000 with steps of 50. The optimal number of hidden nodes was found to be 400. Learning rates ranging from 0.1 to 1.0 were explored in steps of 0.1. The chosen learning rate was a constant learning rate of 0.9, because an adaptive learning rate was observed to decrease AUCs. The L2 penalty was set to the default value of 0.0001.

Random Forest

A random forest (RF) is constructed by combining multiple decision trees during training (Dietterich, 2000a; Ho, 1995; Breiman, 2001). All decision trees in the random forest contribute to the determination of the final output class. The output class is determined by averaging the probabilities produced by the

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

individual trees. The range of number of estimators (decision trees) explored was from 1 to 1000 in steps of 100. The optimal setting was found to be 400. The largest AUC results were obtained when the nodes are expanded until almost all leaves are pure. We tested our model with the maximum depth of the tree ranging from 15 to 25 and found that the maximum AUC was obtained at a depth of 20. All features were used in every tree. To measure the quality of a split we used the default Gini index (Strobl et al., 2007) and the maximum number of features to consider when looking for the best split in a node was set to 2, as calculated by the function tuneRF available in the R package randomForest (version 4.6-12).

Adaptive Boosting

Adaptive Boosting or AdaBoost (AB) was developed for binary classification problems and tweaks "weak learners" by focusing on the instances that were wrongly classified by previous classifiers (Freund and Schapire, 1997). Therefore the training error decreases over the iterations. The additive model of AdaBoost can be formulated as following. The output of each weak learner is described by:

$$L_K(x) = \sum_{k=1}^{K} l_k(x).$$

where K is the total number of iterations and $l_k(x)$ is the output function of the weak learner when taking the instance x as input. To minimize the training error E_k for each iteration k, AdaBoost uses:

$$E_k = \sum_{i=1}^{N} E(L_{k-1}(x_i) + \alpha_k h(x_i)),$$

where $h(x_i)$ is the predicted output of a weak learner for every instance x_i in the training set, α_k is the assigned coefficient that minimizes the training error, and N is the total number of instances in the training set.

We used AdaBoost on a random forest (RF) classifier that performed just better than chance on the training data. The optimal parameters of this RF were found to be 100 decision trees (estimators) and a maximum depth of 1. This means all of the trees were decision stumps. The number of estimators was established at 100 after exploring a range from 1 to 1000 estimators with steps of 50. A maximum depth of 1 was chosen because AdaBoost is known to perform better with decision stumps (Ridgeway, 1999).

Gradient Boosting

In gradient boosting (GB) an initial poor fit on the data is improved by fitting base-learners (e.g. decision trees) to the negative gradient of a specified loss function (Friedman, 2001). Gradient boosting can be described by:

$$\hat{f} = argmin_f E_{x,y}[\rho(Y, f(X)],$$

where $X = \{x_1, ..., x_n\}$ and $Y = \{y_1, ..., y_n\}$, forming the training set $\{(x_1, y_1), ..., (x_n, y_n)\}$. $\hat{f}$ minimizes expectation E of the loss function ρ over all prediction functions f that take X as input.

We used gradient boosting on 50 estimators (decision trees) with a maximum depth of 15. We established the number of estimators by exploring a range of 1 to 1000 estimators with steps of 50. We tested our model with the same maximum depth of the tree as for the decision tree classifiers. We then gradually decreased the maximum depth taking steps of 1, arriving at 15 as the best setting. The minimum number of samples at a leaf node was set to 5, as this was the number found to maximize AUC. Stochastic gradient boosting was performed with a subsampling of 0.9.

Performance Assessment

Model performance was assessed in terms of AUC and precision at different recall rates (10%, 40% and 60% recall was used). As the classifiers used construct models stochastically, five training runs were carried out for each of the 20 models (five machine learning approaches times four training sets). The five training runs were done after optimizing the classifiers' parameters with LOO CV. Models were evaluated on five validation sets. Each validation set corresponds to data from one bacterial species. Data of *R. capsulatus*, *S. pyogenes* and *S. enterica* was also used for training, while *E. coli* and *M. tuberculosis* data was used exclusively for validating the models (Table 1). The species for validation were chosen to be one species of the same taxa as and one of a different taxa from the species used for training. Median, mean and standard deviation of the performance measurements across the five training runs were calculated.

Additionally, to highlight the difference in performance between the models, we used a "winner-gets-all" comparison by ranking the methods based on their precision at different recall rates for each

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

validation set. The model(s) with the highest precision at a given recall for a specific validation set were ranked 1 for that validation set. Ties were all given the same rank. At the end of the ranking process, each model has 15 ranks corresponding to one rank per validation set $\times$ recall rate combination.

Statistical significance of the difference in performance between models was estimated using a pair-wise Wilcoxon signed rank sum (also called Mann-Whitney) tests on precision vectors, and p-values were corrected for multiple comparison using False Discovery Rate (FDR). The training data and the classifier used were considered factors to group the models. Analysis of variance (ANOVA) was performed to explore the effects of classifier and training data on the precision values, and the Tukey's Honest Significant Difference (HSD) (Tukey, 1949) method was used to asses the significance on the differences between the mean precision of classifiers, training data, and models. All statistical analyses were carried out using R (version 3.4.1).

Attribute Importance

To gain insight on how important each attribute is in inferring whether or not a sequence encodes a bona fide sRNA, we used the function `varImp` available in the R package randomForest (version 4.6-12). To use this function, we first created a RF classifier using the `randomForest` function with `ntree` set to 400 and `mtry` set to 2. These were the optimal parameters found when tuning the RF classifier (see above). We generated the RF model using the combined training data (Table 1). Attribute importance was measured in terms of the mean decrease in accuracy caused by an attribute during the out of bag error calculation phase of the RF algorithm (Breiman, 2001). The more the accuracy of the RF model decreases due to the exclusion (or permutation) of a single attribute, the more important that attribute is deemed for classifying the data.

RESULTS

In this section models are identified by the classifier and the training data used. Training and validation data sets are labelled with the corresponding bacterial species: Ec = *Escherichia coli*, Mt = *Mycobacterium tuberculosis*, Se = *Salmonella enterica*, Sp = *Streptococcus pyogenes*, and Rc = *Rhodobacter capsulatus*. AUC scores for all the models per validation set are shown in Fig. 2. Fifteen out of the twenty models have an averaged AUC above 0.75 on all the validation data sets. Only one model (LR-Sp) performed worse than a random classifier on two validation data sets (Mt and Rc). Models generated by LR had lower AUCs than models generated by the other classifiers used. There was low variance of AUC between training runs: standard deviations of the AUCs ranged from 0.00 to 0.05 for all the models.

As validation sets are unbalanced (i.e., there are much more negative instances than positive instances), AUC scores are over-optimistic on the model performance. Thus, we looked at precision values at different recall rates. Fig. 3 shows the distribution of precision values for each classifier at three different recall values. LR models have significantly lower precision values than models obtained by the other four classifiers (p-values $< 2e^{-16}$ as per the Mann-Whitney test and Tukey's HSD test). On the other hand, RF models have significantly higher precision values than models obtained by all other classifiers. Significant differences in precision values among the five classifiers are indicated in Table 2.

ANOVA results indicated that the classifier and the training data are both significant factors to explain variance in precision values (F-statistic $= 118.98$, p-value $< 2e^{-16}$ and F-statistic $= 19.03$, p-value $4.13e^{-12}$, respectively). A significant interaction between these two factors (F-statistic $= 3.90$, p-value $6.46e^{-6}$) was also found by ANOVA. Models trained on the Rc training data have significantly lower precision values than models trained on the other three training sets (p-values $< 5e^{-6}$ as per the Mann-Whitney test and the Tukey's HSD test). According to the Mann-Whitney test, models trained on the Sp data have significantly lower precision values than models trained on the Se training data or on the combined data (p-values $< 5e^{-5}$).

The standard deviations of the precision values was higher than those of the AUCs. At 10% recall, the standard deviation of the precision values across all models varied from 0.00 to 0.39 with a mean standard deviation of 0.06. At 40% recall, the standard deviation of the precision values across all models varied from 0.00 to 0.21 with a mean standard deviation of 0.03. At 60% recall, the standard deviation of the precision values ranged from $4.77e^{-5}$ to 0.17 with a mean standard deviation of 0.03. The classifiers producing the most variable models were MP and GB (Figs. 3 and 4) with average standard deviations above the overall mean standard deviation. For example, MP and GB models have an average standard deviation of the precision values at 40% recall of 0.056 and 0.051, respectively.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

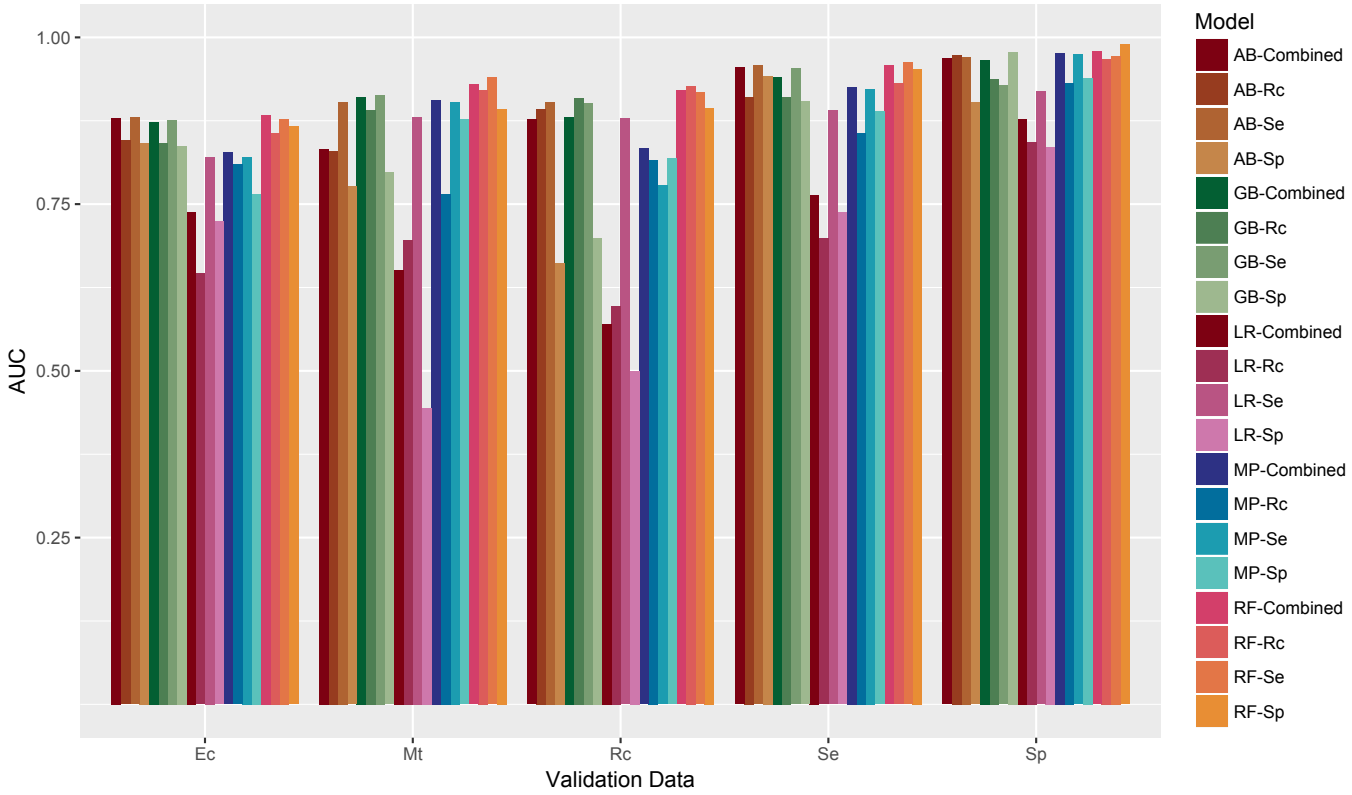

Figure 2. Average Area under the ROC curve (AUC) per model on each validation data set. Models are colour coded by the classifier used to generate them: AB = Adaptive Boosting (brown gradient), GB = Gradient Boosting (green gradient), LR = Logistic Regression (pink gradient), MP = Multilayer Perceptron (blue gradient), RF = Random Forest (red gradient). Training and validation data sets are labelled with the corresponding bacterial species: Ec = *Escherichia coli*, Mt = *Mycobacterium tuberculosis*, Se = *Salmonella enterica*, Sp = *Streptococcus pyogenes*, and Rc = *Rhodobacter capsulatus*. The combined data is the training data of *S. enterica*, *S. pyogenes* and *R. capsulatus* together. Error bars are not plotted as the range of the standard deviations across all models is 0.00 to 0.05.

Table 2. Pair-wise statistically significant differences in precision values between classifiers (AB = Adaptive Boosting, GB = Gradient Boosting, LR= Logistic Regression, MP = Multilayer Perceptron, RF = Random Forest). Acronyms in the cells indicate that a given row classifier has significantly lower precision values (p-values < 0.005) than a column classifier according to the Tukey's HSD test and/or Mann-Whitney test (MW).

	MP	GB	AB	RF
LR	Tukey's HSD / MW	Tukey's HSD / MW	Tukey's HSD / MW	Tukey's HSD / MW
MP		MW	Tukey's HSD / MW	Tukey's HSD / MW
GB			MW	Tukey's HSD / MW
AB				MW

To emphasize differences in performance among the models, we ranked each model based on the precision values obtained on each validation set at three fixed recall rates. Ties were assigned the same rank. As LR was clearly outperformed by the other four classifiers, we excluded LR results from this analysis. Fig. 4 depicts the mean rank of the models obtained by each classifier as a function of the interaction between classifier and training set used. AB is the classifier least susceptible to variations in rank due to the training data; while, MP is the classifier with more variation in rank due to the training data (Fig. 4).

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

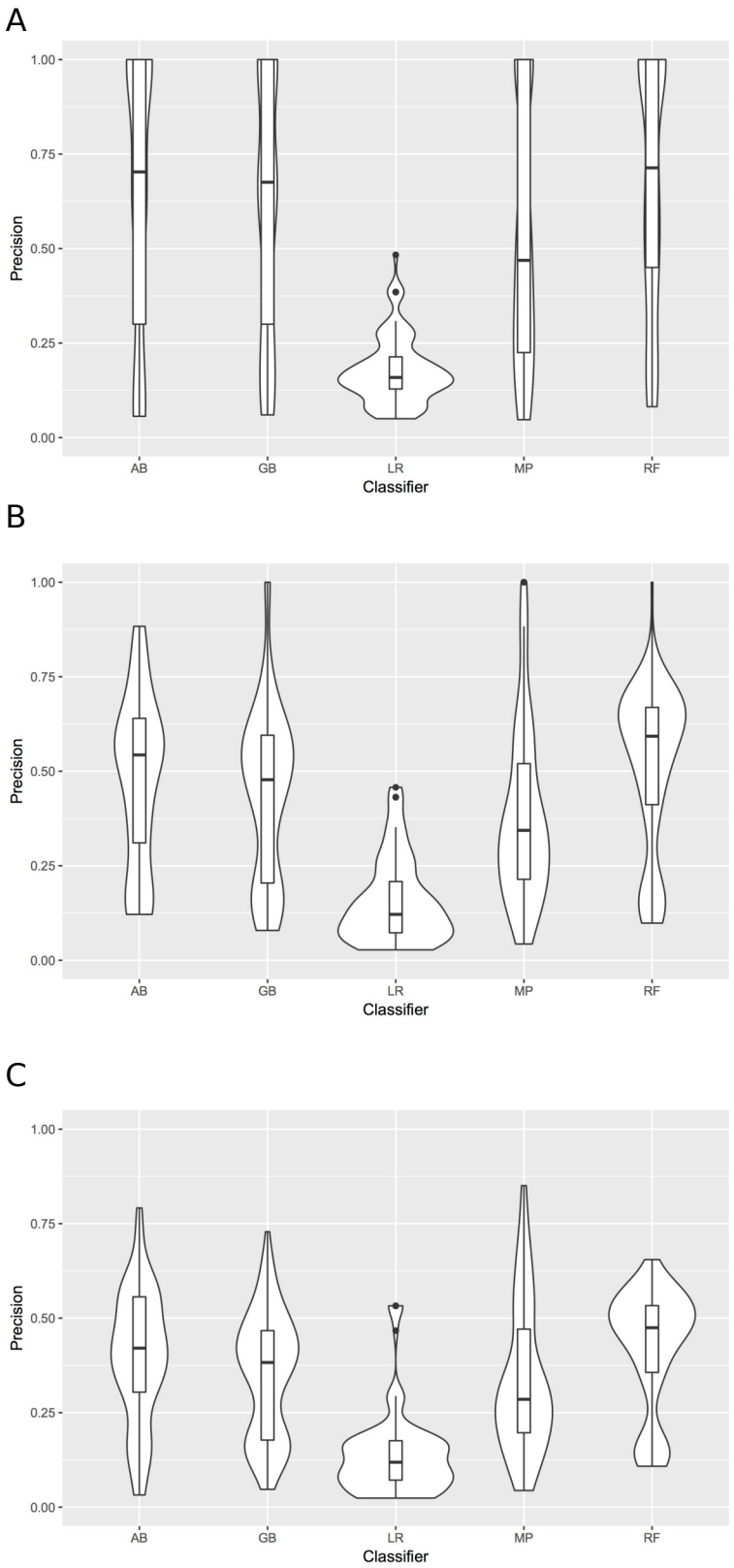

Figure 3. Distribution of precision values at different recall rates per classifier. A. Violin plot illustrating the distribution of precision values at 10% recall for all models obtained with each classifier. Inside the distribution shape a box indicates the range from the 25 percentile to 75 percentile of the precision values. B. Same as A, but at 40% recall. C. Same as A, but at 60% recall. AB = Adaptive Boosting, GB = Gradient Boosting, LR = Logistic Regression, MP = Multilayer Perceptron, RF = Random Forest.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

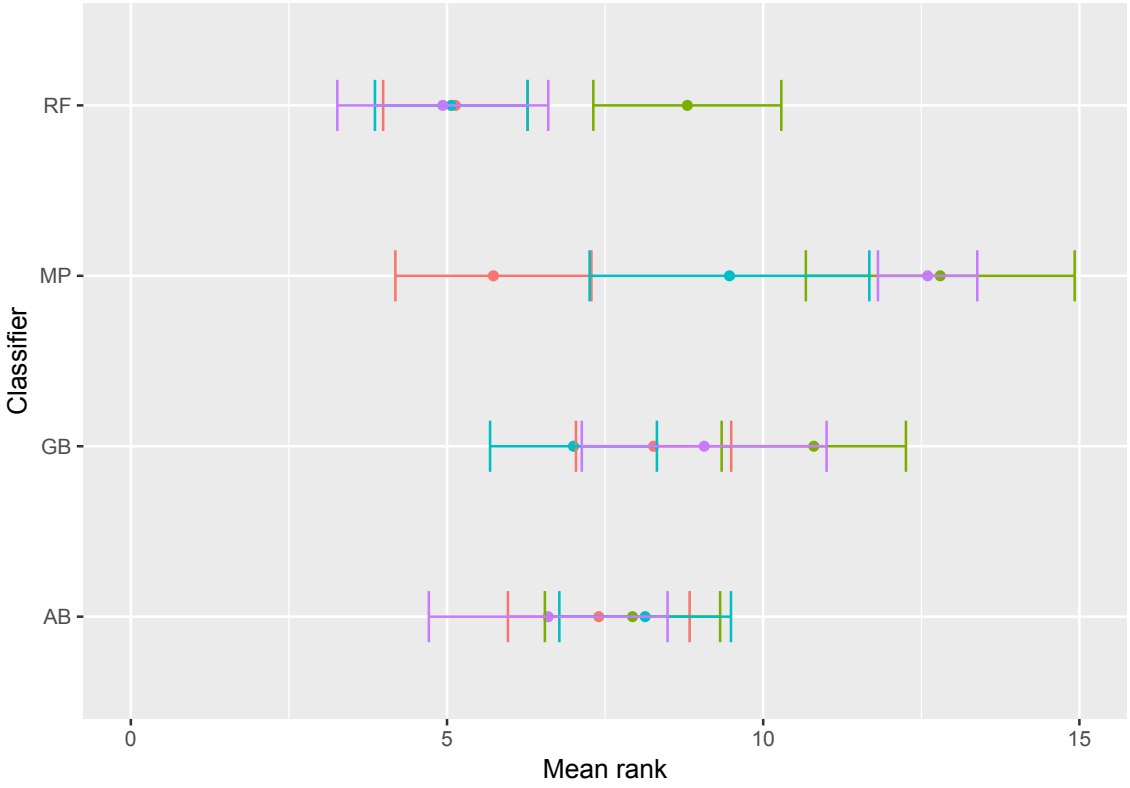

Figure 4. Effect of training data on classifier mean rank. The average rank of the models obtained with each classifier is depicted as a function of the training data used to create the model. The dot represents the mean rank and bars represent standard error. Colour indicates the training data used: Red = Combined data, Green = *R. capsulatus* data, Blue = *S. enterica* data, Purple = *S. pyogenes* data. Classifiers are indicated by AB = Adaptive Boosting, GB = Gradient Boosting, MP = Multilayer Perceptron, RF = Random Forest.

The best performing models (in terms of rank and precision values) were RF-Se, RF-Sp and RF-Combined. These three models obtained significantly higher precision values (p-values < 0.05, Mann-Whitney test) than all other models but the MP-Combined model and the AB-Sp model. Fig. 5 shows the precision-recall curves of these five models (RF-Se, RF-Sp, RF-Combined, AB-Sp, and MP-Combined) on the validation data sets. These five models can be considered as comparable in terms of precision values at different recall rates. To facilitate other researchers to rank their own sRNAs, we have created sRNARanking, an R script that produces the predictions generated by the RF-Combined model. sRNARanking takes as input the feature table produced by sRNACharP and calculates the probability of being a bona fide sRNA for each sRNA included in the feature table. sRNARanking is available at `https://github.com/BioinformaticsLabAtMUN/sRNARanking`.

Based on the mean decrease in accuracy estimated by the random forest algorithm, all attributes contribute positively to obtain a more accurate model (Fig. 6). The seven attributes clustered in three levels of importance: those with a mean decrease in accuracy greater than 20; those with a mean decrease in accuracy between 10 and 15, and those with a mean decrease in accuracy lower than 10. The most important attributes are the distance to the closest ORFs and the distance to the closest predicted rho-independent terminator. The two attributes that seem to contribute the least to the accuracy of a model are the Boolean features indicating whether or not a genomic sequence is transcribed on the same strand as its closest ORFs.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

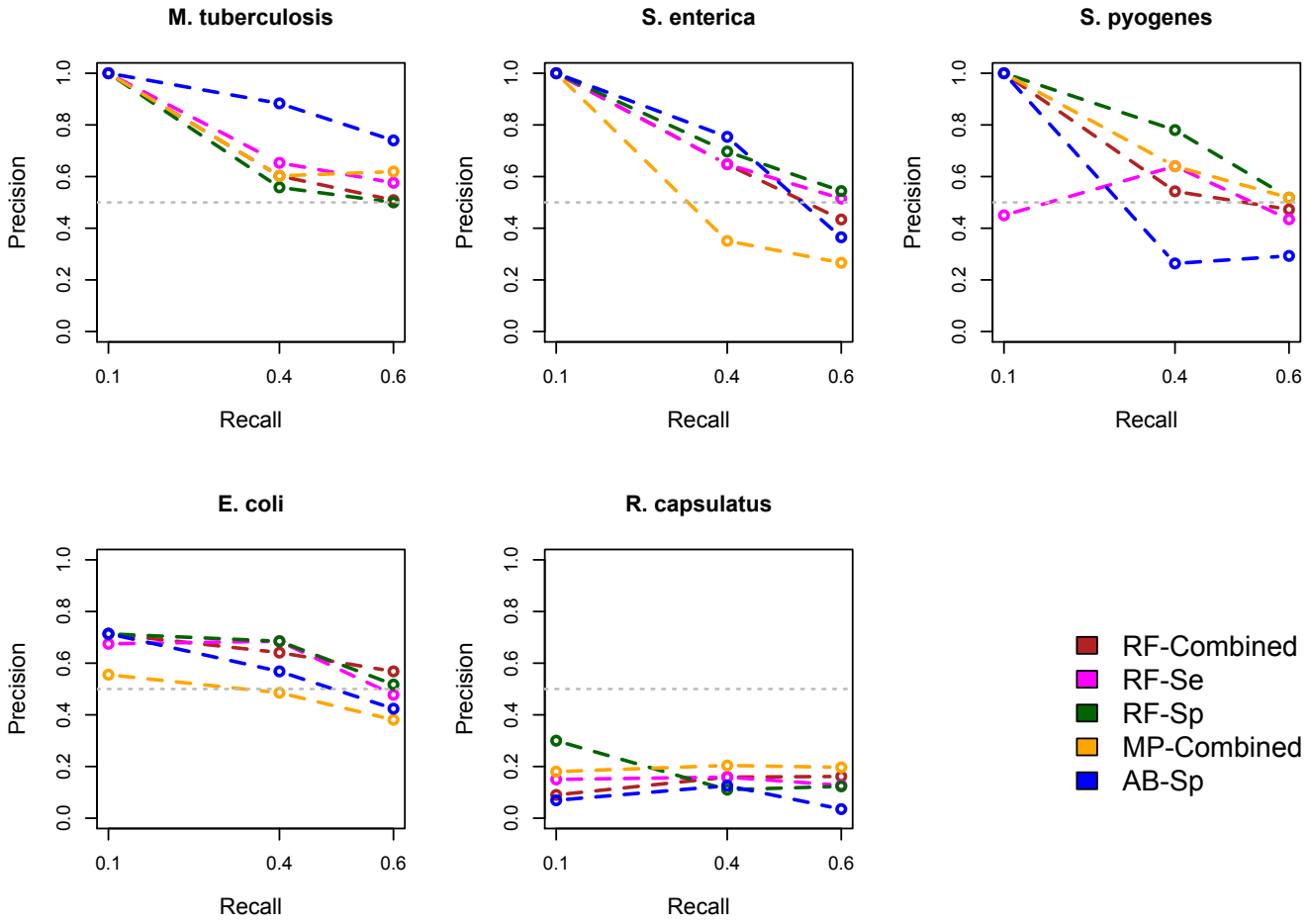

Figure 5. Precision-Recall curves of the best performing models on each validation set. Median precision values across the five training runs are shown at 10%, 40% and 60% recall rate. Training and validation data sets are labelled with the corresponding bacterial species: Ec = *Escherichia coli*, Mt = *Mycobacterium tuberculosis*, Se = *Salmonella enterica*, Sp = *Streptococcus pyogenes*, and Rc = *Rhodobacter capsulatus*. The combined data is the training data of *S. enterica*, *S. pyogenes* and *R. capsulatus* together. The horizontal grey line is drawn at 0.5 precision.

DISCUSSION

We believe that the distances to the closest ORFs are the most important attributes partially due to a bias in the training data. 93% of the negative instances (random genomic sequences) in the combined training data overlap the two neighbouring ORFs (i.e., their distances to their closest ORFs are zero), while 70% of the positive instances (bona fide sRNAs) are intergenic (i.e., their absolute distances to their closest ORFs are greater than zero). This bias in the data may be corrected as more antisense sRNAs (asRNAs) and partially overlapping sRNAs are experimentally verified as bona fide sRNAs.

We hypothesize that *R. capsulatus* training data produced worse performing models because it includes as positive instances a higher number of non-intergenic sRNAs (18 or 50%). In fact, the best performing models obtained consistently lower precision values for *R. capsulatus* and *E. coli* validation data sets (Fig. 5). These two bacterial species have the higher proportion of non-intergenic bona fide sRNAs: 51% and 40% of the bona fide sRNAs of *R. capsulatus* and *E. coli*, respectively, overlap neighbouring ORFs; while 17.4%, 26.5% and 36.8% of the bona fide sRNAs of *S. pyogenes*, *S. enterica* and *M. tuberculosis*, respectively, overlap neighbouring ORFs. Additionally, 17 *R. capsulatus* putative sRNAs included as positive instances were found to be conserved in the genome of at least two other bacterial species but

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

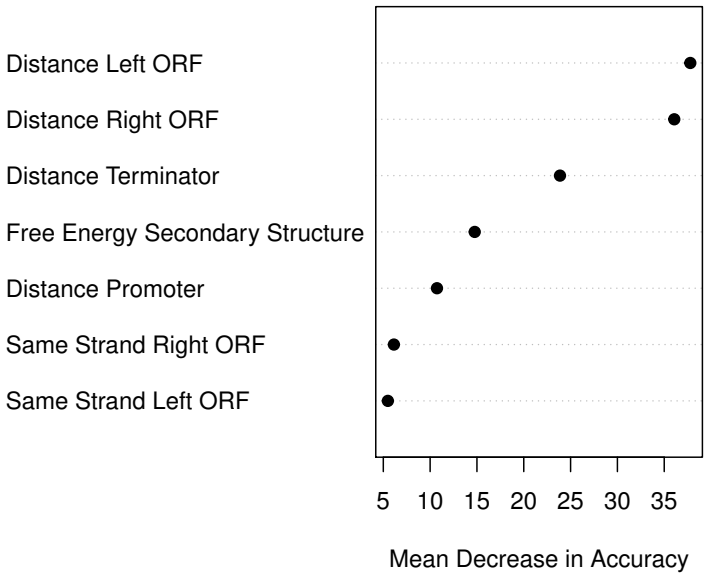

Figure 6. Attribute importance. Mean decrease in accuracy per attribute as estimated by the random forest algorithm. Attribute importance is plotted on the x-axis. Attributes are ordered top-to-bottom as most- to least-important. Three levels of importance are observed: high importance attributes (distances to closest ORFs and distance to terminator); medium importance attributes (free energy of secondary structure and distance to promoter), and low importance attributes (same strandness as closest ORFs).

have not been verified in the wet lab. Some of these 17 putative *R. capsulatus* sRNAs chosen as positive instances based on sequence conservation may actually be false positives.

With respect to the different machine learning approaches assessed, RF seems to be better suited for the task of prioritizing bona fide sRNAs than the other four classifiers (AB, GB, MP and LR). To be able to use deep learning for sRNA prioritization, data sets at least one order of magnitude larger than the ones currently available are required.

To demonstrate the ability of the models to generalize to other bacterial species, we validated the models on data from bacterial species that were not part of the training set. In fact, using data from the same bacterial species on the training and validation sets was not a factor to explain variance in model performance. This indicates that models are able to learn sRNAs features that are species independent, and even taxa independent as the precision values obtained in the *M. tuberculosis* validation set suggest (Fig. 5). Using data from different bacterial species and experimental conditions is expected to lead to improved predictive models. In fact, training the classifiers with the combined data generated models that either outperform, or were comparable to, the models obtained from training the classifiers with data from a single bacterial species (Fig. 2 and Fig. 4). To allow other researchers to rank their own sRNAs, we have implemented sRNARanking, an R script containing the RF-Combined model.

CONCLUSION

A multitude of sRNAs have been detected in many bacterial species. The sheer number of novel putative sRNAs reported in the literature makes it infeasible to validate in the web lab each of them. Thus, there is the need for computational approaches to characterize putative sRNAs and to rank these sRNAs on the basis of their likelihood of being bona fide sRNAs. In this study we have applied five machine learning approaches to obtain models for predicting whether or not a given genomic sequence (represented with seven numerical attributes) encodes a bona fide sRNA. Attributes were chosen based on the feasibility of calculating them computationally while only requiring the sRNA and genome sequences, and a genome

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

annotation file. The most important attributes are the distance to the closest ORFs and the distance to the closest predicted rho-independent terminator. To enable other researchers to easily obtain these seven features for their own putative sRNAs, we have developed sRNACharP.

We used five machine learning methods and four different training sets which produced twenty models to rank putative sRNAs on the basis of their likelihood of being bona fide sRNAs. The best performing models were obtained with RF; while LR models behaved less effectively. To assess the ability of the models to generalize to other bacterial species, we validated the models in data from bacterial species that were not part of the training set. Our results demonstrate that machine learning approaches are indeed able to detect intrinsic features of sRNAs common to a number of bacterial species, overcoming the challenge of the low sequence conservation of sRNAs. As the number of detected sRNAs continues to raise, computational predictive models as the ones here generated will become increasingly valuable to guide further investigations.

ABBREVIATIONS

LR: logistic regression; MP: multilayer perceptron; AB: adaptive boosting; GB: gradient boosting; RF: random forest; FDR: false discovery rate; AUC: area under receiver operating characteristic curve; LOO CV: leave-one-out cross-validation; ORF: open reading frame; nts: nucleotides; sRNA: small non-coding RNA.

ACKNOWLEDGEMENTS

We thank Emilio Palumbo for providing technical support for the implementation in Nextflow, and Dr. Meruvia-Pastor for providing feedback on the manuscript.

REFERENCES

Bishop, C. M. (1995). Neural networks for pattern recognition. *Oxford university press.*, 3rd ed.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society Series B (Methodological)*, pages 215–242.

Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., and Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nat Biotechnol*, 35(4):316–319.

Di Tommaso, P., Palumbo, E., Chatzou, M., Prieto, P., Heuer, M. L., and Notredame, C. (2015). The impact of docker containers on the performance of genomic pipelines. *PeerJ*, 3:e1273.

Dietterich, T. (2000a). An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157.

Dietterich, T. G. (2000b). Ensemble methods in machine learning. *Multiple classifier systems*, 1857:1–15.

Fahlman, S. E. (1988). Faster-learning variations on backpropagation: An empirical study. *Proceedings of the Connectionist Models Summer School*, pages 38–51.

Freund, Y. and Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232.

Gama-Castro, S., Salgado, H., Santos-Zavaleta, A., Ledezma-Tejeida, D., Muñiz-Rascado, L., García-Sotelo, J. S., Alquicira-Hernández, K., Martínez-Flores, I., Pannier, L., Castro-Mondragón, J. A., Medina-Rivera, A., Solano-Lira, H., Bonavides-Martínez, C., Pérez-Rueda, E., Alquicira-Hernández, S., Porrón-Sotelo, L., López-Fuentes, A., Hernández-Koutoucheva, A., Del Moral-Chávez, V., Rinaldi, F., and Collado-Vides, J. (2016). RegulonDB version 9.0: high-level integration of gene regulation, coexpression, motif clustering and beyond. *Nucleic Acids Res*, 44(D1):D133–43.

Gómez-Lozano, M., Marvig, R. L., Molina-Santiago, C., Tribelli, P. M., Ramos, J.-L., and Molin, S. (2015). Diversity of small RNAs expressed in *Pseudomonas* species. *Environ Microbiol Rep*, 7(2):227–36.

Grüll, M. P., Peña-Castillo, L., Mulligan, M. E., and Lang, A. S. (2017). Genome-wide identification and characterization of small RNAs in *Rhodobacter capsulatus* and identification of small RNAs affected by loss of the response regulator CtrA. *RNA Biol*, pages 1–12.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018

Hamada, M., Kiryu, H., Sato, K., Mituyama, T., and Asai, K. (2009). Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics (Oxford, England)*, 25(4):465–473.

Ho, T. K. (1995). Random decision forests. *Document Analysis and Recognition*, 1:278–282.

Kerlirzin, P. and Vallet, F. (1993). Robustness in multilayer perceptrons. *Neural computation*, 5(3):473–482.

Kingsford, C. L., Ayanbule, K., and Salzberg, S. L. (2007). Rapid, accurate, computational discovery of Rho-independent transcription terminators illuminates their relationship to DNA uptake. *Genome Biol*, 8(2):R22.

Kröger, C., Dillon, S. C., Cameron, A. D. S., Papenfort, K., Sivasankaran, S. K., Hokamp, K., Chao, Y., Sittka, A., Hébrard, M., Händler, K., Colgan, A., Leekitcharoenphon, P., Langridge, G. C., Lohan, A. J., Loftus, B., Lucchini, S., Ussery, D. W., Dorman, C. J., Thomson, N. R., Vogel, J., and Hinton, J. C. D. (2012). The transcriptional landscape and small RNAs of *Salmonella enterica* serovar typhimurium. *Proc Natl Acad Sci U S A*, 109(20):E1277–86.

Le Rhun, A., Beer, Y. Y., Reimegård, J., Chylinski, K., and Charpentier, E. (2016). RNA sequencing uncovers antisense RNAs and novel small RNAs in *Streptococcus pyogenes*. *RNA Biol*, 13(2):177–95.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomForest. *R news*, 2(3):18–22.

McClure, R., Tjaden, B., and Genco, C. (2014). Identification of sRNAs expressed by the human pathogen *Neisseria gonorrhoeae* under disparate growth conditions. *Frontiers in microbiology*, 5:456.

Michaux, C., Verneuil, N., Hartke, A., and Giard, J.-C. (2014). Physiological roles of small RNA molecules. *Microbiology*, 160(Pt 6):1007–19.

Miotto, P., Forti, F., Ambrosi, A., Pellin, D., Veiga, D. F., Balazsi, G., Gennaro, M. L., Di Serio, C., Ghisotti, D., and Cirillo, D. M. (2012). Genome-wide discovery of small RNAs in *Mycobacterium tuberculosis*. *PLoS One*, 7(12):e51950.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Quinlan, A. R. and Hall, I. M. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26(6):841–2.

Ridgeway, G. (1999). The state of boosting. *Computing Science and Statistics*, pages 172–181.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2):197–227.

Solovyev, V. V. and Salamov, A. (2011). Automatic annotation of microbial genomes and metagenomic sequences. *Metagenomics and its Applications in Agriculture, Biomedicine and Environmental Studies*.

Storz, G., Vogel, J., and Wassarman, K. M. (2011). Regulation by small RNAs in bacteria: expanding frontiers. *Mol Cell*, 43(6):880–91.

Strobl, C., Boulesteix, A. L., and Augustin, T. (2007). Unbiased split selection for classification trees based on the Gini index. *Computational Statistics and Data Analysis*, 52(1):483–501.

Thomason, M. K., Bischler, T., Eisenbart, S. K., Forstner, K. U., Zhang, A., Herbig, A., Nieselt, K., Sharma, C. M., and Storz, G. (2015). Global transcriptional start site mapping using differential RNA sequencing reveals novel antisense RNAs in *Escherichia coli*. *Journal of Bacteriology*, 197(1):18–28.

Tukey, J. W. (1949). Comparing individual means in the analysis of variance. *Biometrics*, 5(2):99–114.

UCSC website (2018). BED format description.

Walker, S. H. and Duncan, D. B. (1967). Estimation of the probability of an event as a function of several independent variables. *Biometrika*, 54(1-2):167–179.

Zeng, Q. and Sundin, G. W. (2014). Genome-wide identification of Hfq-regulated small RNAs in the fire blight pathogen *Erwinia amylovora* discovered small RNAs with virulence regulatory function. *BMC genomics*, 15:414–2164–15–414.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26974v1 | CC BY 4.0 Open Access | rec: 2 Jun 2018, publ: 2 Jun 2018