PhyMet^2: MethanoGram

Supplementary Materials 1

Michał Burdukiewicz, Przemysław Gagat, Sławomir Jabłoński,
Jarosław Chilimoniuk, Michał Gaworski, Paweł Mackiewicz, Marcin Łukaszewicz

Introduction

In contrast to animals and plants, microorganisms are usually grown in the form of a colony, from a single cells, in order to be studied in detail, e.g. in terms of growth conditions or morphology. From metagenomics analyses, we know that there is a plethora of new uncultivated microorganisms, including methanogens. Unfortunately, we can often cultivate much less species than actually occur in a given environment [2]. Therefore we have created MethanoGram, a predictor of culturing conditions of methanogens. Using random forests trained on the selected subsequences (n-grams) of 16S rRNA , MethanoGram is able to estimate:

- growth doubling time,
- optimal growth temperature,
- optimal growth pH,
- optimal growth NaCl.

Here we describe the process of tuning and evaluating of set of classifiers constituting MethanoGram. Scripts necessary to reproduce training and evaluation of MethanoGram are available on GitHub: https://github.com/michbur/PhyMet2_supplements.

Tuning and evaluating of MethanoGram

Datasets

In order to train MethanoGram, we used n-grams, i.e. subsequences of the length n that were extracted from 16S rRNA and *mcrA* nucleotide sequences deposited in the PhyMet^2 database. We chose only those species that have known 16S rRNA and *mcrA* sequences as well as all important culturing conditions: growth doubling time, optimal growth temperature, optimal growth pH and optimal growth NaCl.

We removed all sequences containing unknown nucleotides (B, D, K, M, N, R, S, V, W, Y). The final set included 67 methanogens (Fig.1).

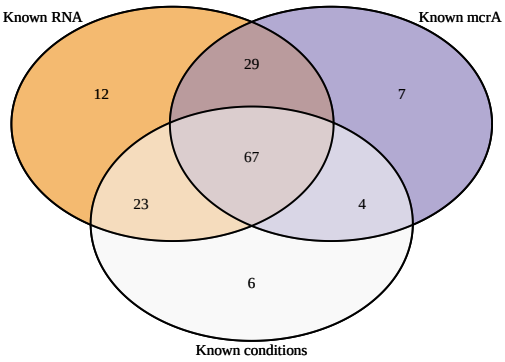

Figure 1: The Venn diagram of methanogenic species used in the analysis.

Tuning procedure

To estimate the culturing conditions we chose the random forests algorithm because of its high performance and resistance to over-fitting [1]. We used random forests implemented in the **ranger** R package [3]. In order to find the optimal values of hyperparameters, we performed a nested cross-validation of random forest classifiers (Fig.2). The inner loop was 5-fold cross-validation and the outer loop was more demanding 3-fold cross-validation.

We have optimized two hyperparameters related to the random forest algorithm: the number of variables sampled for splitting at each node, the number of trees in the forest and the minimal node size. In the tuning procedure we also incorporated different data sources, levels of feature selection and n-gram lengths. In total we tested 5184 combinations of various parameters and data sets.

The length and source of n-grams

We considered continuous 1-, 2-, 3-, 4-, 5- and 6-grams. The number of possible n-grams for a nucleotide sequence is equal to 4^n, so the number of features (possible n-grams) ranges from 4 for 1-grams to 4096 for 6-grams. Since further increase in the n-gram length did not significantly decrease the error of algorithm, we did not include the longer n-grams in the final analyses.

The algorithm was trained on n-grams extracted from:

- 16S rRNA,
- mcrA,
- 16S rRNA and mcrA.

In the third case, if the same n-grams were found in the sequences of these two genes, they were treated as two different features and were labeled by their gene source.

Feature selection

To select the most informative n-grams, we used Pearson's correlation between the fraction of the given n-gram and the parameter describing the culture condition. The correlation coefficient was calculated by *correls* function from the Rfast package in R environment. We selected top 10%, 25% and 50% features characterized by the highest correlation coefficient. The strictest feature selection, i.e. 10% and 25% proved to create the most efficient classifiers.

The approach assuming all n-grams, i.e. without any feature selection, showed the worst performance (results not shown).

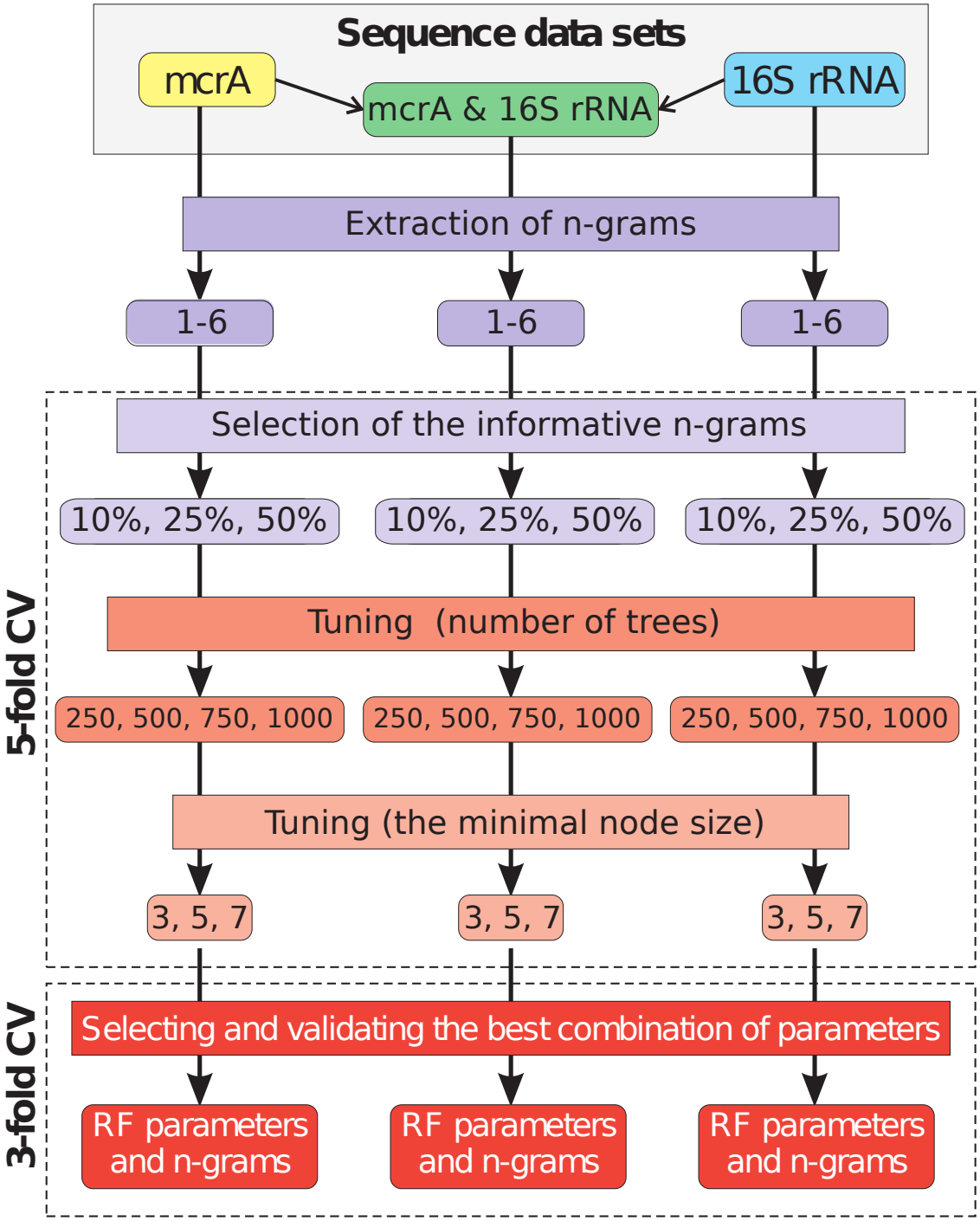

Figure 2: The scheme of tuning procedure which was aimed at receiving the optimal values and combination of random forest parameters and the informative n-grams. The tunning of parameters in the random forest algorithm was subjected to 5-fold cross-validation and the final step to 3-fold cross-validation.

Minimal node size

In order to set the value for the minimal node size, we assumed aside from the optimal number of variables, as advised by the literature, i.e. 5, also 3 and 7 variables. There were no visible patterns in the optimal value of the minimum node size, besides the fact that the value proposed by the literature was rarely producing the best-performing predictors.

Results of tuning

Table 1: Mean errors of the best predictors found in the nested cross-validation for three possible data sources.

Culture condition	16S rRNA and mcrA	Only 16S rRNA	Only mcrA
Growth doubling time [h]	1.6780	1.6817	1.9704
Optimal temperature [°C]	1.0266	1.0274	1.0395
Optimal NaCl [mol/dm^3]	0.1980	0.1980	0.2176
Optimal pH	0.6053	0.6053	0.6218

We discovered that predictors employing both data from mcrA and 16S rRNA have the lowest mean error. Nevertheless, we found out that predictors based solely on 16S rRNA have almost the same mean error (Table 1). Therefore we decided to train MethanoGram only on 16S rRNA sequences. This way to predict culture conditions, a user needs only 16S rRNA sequence of newly discovered methanogen.

The best combinations of parameters for each condition are presented in Table 2.

Table 2: The hyperparameters found in the nested cross-validation.

Culture condition	n-grams (16S rRNA)	Fraction of n-grams	Logarithm	The number of trees	Minimal node size	Mean error
Growth doubling time [h]	1 and 6	10%	yes	500	3	16.30
Optimal temperature [°C]	1 and 3	25%	yes	250	5	6.67
Optimal NaCl [mol/dm^3]	4 and 6	10%	no	250	5	0.13
Optimal pH	1 and 4	25%	no	500	7	0.13

References

[1] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, October 2001.

[2] Jörg Overmann, Birte Abt, and Johannes Sikorski. Present and Future of Culturing Bacteria. *Annual Review of Microbiology*, 71(1):711–730, 2017.

[3] Marvin N. Wright and Andreas Ziegler. Ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. *arXiv:1508.04409 [stat]*, August 2015.