

Gene Regulatory Network Inference Using Ensembles of Predictors

Sergio Peignier, Baptiste Sorin, Federica Calevro

▶ **To cite this version:**

Sergio Peignier, Baptiste Sorin, Federica Calevro. Gene Regulatory Network Inference Using Ensembles of Predictors. 33rd IEEE International Conference on Tools with Artificial Intelligence, IEEE, Nov 2021, Virtual event, United States. hal-03022606v2

HAL Id: hal-03022606
https://hal.science/hal-03022606v2

Submitted on 8 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Ensemble Learning Based Gene Regulatory Network Inference

Sergio Peignier[1,a], Baptiste Sorin[1,b], and Federica Calevro[1,c]

[1] *Univ Lyon, INSA Lyon, INRAE , BF2I, UMR0203, F-69621*, Villeurbanne, France
[a]sergio.peignier@insa-lyon.fr
[b]baptiste.sorin@insa-lyon.fr
[c]federica.calevro@insa-lyon.fr

Abstract

In the machine learning field, the technique known as ensemble learning aims at combining different base learners in order to increase the quality and the robustness of the predictions. Indeed, this approach has widely been applied to tackle, with success, real world problems from different domains, including computational biology. Nevertheless, despite their potential, ensembles combining results from different base learners, have been understudied in the context of gene regulatory network inference. In this paper we applied genetic algorithms and frequent itemset mining, to design small but effective ensembles of gene regulatory network inference methods. These ensembles, were evaluated and compared to well-established single and ensemble methods, on real and synthetic datasets. Results showed that small ensembles, consisting of few but diverse base learners, enhance the exploration of the solution space, and compensate base-learners biases, outperforming state-of-the-art methods. Results advocate for the use of such methods as gene regulatory network inference tools.

Keywords— Bioinformatics, Gene Regulatory Network Inference, Ensemble Learning

1 Introduction

Ensemble learning is a machine learning technique, that combines multiple algorithms, with the aim of leading to better predictive performances than its constituent algorithms [1, 2]. This approach has been used successfully to deal with complex real world problems, from different domains, and thus ensemble learning is recognized as a cutting-edge technique, and it has received interest

from the machine learning research community [1, 2]. According to [3], ensemble learning methods have been used increasingly by the computational biology community, to tackle different tasks such as gene expression analysis, and gene interaction identification. Indeed, according to these studies, such technique allows to deal effectively with common problems from the computational biology domain, such as high-dimensional data, and small sample sizes.

An important and challenging task addressed by the systems biology community, consists in reverse-engineering Gene Regulatory Networks (GRNs) [4], i.e. complex regulatory interactions between regulators, being transcription factors (TFs), and their target genes (TGs). Indeed, this is an important challenge, since the control exert by GRNs on the gene expression, is responsible, to an important extent, for important biological mechanisms, such as organogenesis, development, cell-death and the adaption to changing environmental conditions [5]. Given the importance of this task, the advent of high-throughput technologies (RNAseq, Microarray) has motivated the development of several families of algorithms that aim at inferring GRNs from high-throughput data [4]. Each family of methods has its own advantages and drawbacks, each being inclined to reveal some particular types of regulatory interactions [6].

In order to overcome the inner biases of individual methods, some previous works [6] have combined many available GRN inference tools to form a large ensemble, and obtained better and more robust results, across different datasets. Robustness is particularly valuable in real world applications, since it is not straightforward to determine *a priori* which method should be used to analyze a dataset, given the differences existing between organisms and the experimental conditions. Nevertheless, large ensembles incorporating as many inference methods as possible, are likely to require overwhelming computational resources, and including some methods may not be beneficial and could even be detrimental. Therefore, investigating the design of smaller and efficient ensembles of GRN inference methods, seems an interesting research path, that has been under-studied by the computational biology community. In order to tackle this research question, in this work, we used a methodology based on a genetic algorithm and frequent itemset mining to design small, robust and competitive ensembles. We assessed the effectiveness of our methodology by comparing our ensembles with respect to popular approaches, on benchmark datasets described in [6]. Our study shows that our ensembles, are robust and valuable tools for the analysts, specially for real world applications. For the sake of reproducibility, the experiments and the ensembles implementations are available online [1].

The rest of this article is structured as follows. Section 2 describes the related work. Section 3 introduces the GRN inference problem, and describes the methodology, developed in this work, to design suitable ensembles of learners. Section 4 and 5 describe the experimental setup developed in this work and the results, respectively. We conclude with a summary and some perspectives.

[1] https://gitlab.com/bf2i/evening

2 State-of-the-art

2.1 Gene regulatory network inference

Algorithms that aim at reverse-engineering GRNs from gene expression data, have been categorized in three major families [4], as described hereafter.

Model-Based methods infer GRNs by fitting the parameters of a pre-established model, with respect to experimental data [7]. Then, calibrated models allow to simulate and analyze the biological system *in-silico*. Some models, termed *Probabilistic Models* are grounded in probability theory, and they include approaches such as Bayesian networks and Gaussian Graphical Models [4]. Other methods aim at modelling the temporal changes in the expression of genes, through *Dynamical Models*, including Boolean Networks, Dynamic Bayesian Networks and Ordinal Differential Equations [4].

Data-Driven methods analyze high-throughput datasets, to score the level of *dependency* between each TF and each possible TG [4]. Different measures have been used to score the regulatory links. Some algorithms rely on the assumption that the gene expression of a TG and its TF should be correlated, and use *correlation* statistics or more sophisticated information theory scores such as *Mutual Information*, to score regulatory links. Other algorithms are based on feature importance scores assigned by algorithms that are trained to predict the levels of expression of a TG from those of TFs. In practice, these approaches have mostly used *regression* algorithms [4], but recently *classification* algorithms have also been applied successfully [8].

Multi-Network methods infer GRNs by considering heterogeneous sources of data simultaneously [4]. Indeed, besides using gene expression data, these methods also rely on TF binding site patterns, or Chromatin Immuno-Precipitation data. For example the so-called SCENIC method [9], analyzes TF binding site motifs, in order to refine the results produced by the GENIE3 data-driven method [10].

2.2 Ensemble Learning

Ensemble learning is a recognized machine learning technique, that has been applied in the context of supervised learning (i.e., classification and regression), semi-supervised learning, and unsupervised learning (i.e., feature selection and clustering) [2]. Conceptually, this technique aims at training a set of *base learners*, and then integrating their results, using a voting scheme, to form a consensus solution. In practice, four major kinds of procedures to train a set of learners have been identified by [1]: 1) *Input manipulation*, each learner is trained using a slightly modified version of dataset. 2) *Partitioning*, each learner is trained using different subsets (horizontal partitioning) or subspaces (vertical partitioning) of the original dataset. 3) *Learning algorithm manipulation*, each base

model is trained with a different parameter setting, or a different algorithm. 4) *Ensemble hybridization*, at least two of the former strategies are used at once. Whereas, regarding the integration of the base learners results, two families of techniques were described in [1]: 1) *Weighting methods* combine the individual results by assigning weights to each base model, and applying a voting scheme 2) *Meta-learning methods* feed a meta-learner model with the outputs of the base learners, to produce a final integrated output.

It has been shown, that the performance of an ensemble increases with the diversity, and the efficiency of its base learners [1]. Indeed, the inherent diversity of ensemble approaches leads to a better exploration of the solution space than single learners. Moreover, ensembles of diverse learners can also extend the solution representations beyond the base learners' solution space, leading to more flexible and accurate models. Finally, ensembles have also been used to lessen the impact of well-known problems in machine learning, such as the curse of dimensionality, class imbalance, and over-fitting due to small datasets [1].

2.3 Ensemble learning in bioinformatics and GRN inference

Ensemble methods have been successfully used in many real world applications from different fields, such as image and speech analysis [11], or bioinformatics and medicine [3]. Indeed, as reviewed in [3], ensemble learning has been applied to deal with complex biological problems, such as classifying gene expression datasets, identifying interaction between genes and predicting regulatory elements from DNA or protein sequences.

This technique has also been used to develop GRN inference methods. For instance, GENIE3 [10] and GRNBoost2 [12], are data-driven methods, based on well-known ensemble learning algorithms, i.e., Random Forest regression [13] and Gradient Boosting regression [14], respectively. Similarly, in [8], the authors proposed data-driven methods based on well-known classification ensemble algorithms, namely Random Forest [13], Extremely Randomized Trees [15], Gradient Boosting [14] and AdaBoost [16] . Another method called TIGRESS [17], aims at training an ensemble of sparse linear regressors on noisy versions a gene expression dataset, to infer GRNs. All the previous methods use *input manipulation*, and *partitioning* techniques to create ensembles, but rely on a unique kind of base learner, which may induce method-specific biases in predicting regulatory relationships [6].

In order to overcome this problem, a ranked voting procedure was used in [6], to combine the outputs from 35 methods, that participated in the DREAM5 challenge, forming a large ensemble, which was termed "Community". On average, the Community exhibited better results than its base predictors, and its performance revealed to be robust across all datasets, unlike base methods.

3 Materials and Methods

3.1 Overview

In order to build ensembles of methods that are robust across datasets, a naive solution would consist in running as many independent methods as possible and then integrating their results. Nevertheless, including blindly more and more methods, massively increases the computational requirements, without ensuring gains in terms of quality.

In this work, we decided to explore small combinations of methods that lead to suitable and robust results. To do so, we used a genetic algorithm to explore the space of ensembles (i.e., combinations of base learners), and select a population of ensembles that exhibit high inference qualities. Then, we conducted a frequent itemset mining exploration to identify small subsets of base methods, that are frequently selected together by the genetic algorithm, to form suitable ensembles. This analysis aims at discovering, understanding and then exploiting underlying principles that would allow us to combine base-learners to build efficient and robust ensembles.

3.2 Definitions

Gene expression dataset Let a matrix $X \in \mathbb{R}^{I \times J}$ denote a gene expression dataset. The expression of *gene* i in *condition* j is $X_{i,j}$, while $X_{i,\cdot}$ (resp. $X_{\cdot,j}$) represents the vector of levels of expression of *gene* i (resp. *condition* j) for all conditions (resp. genes). The number of genes (rows) and conditions (columns) in X, are denoted I and J.

Gene Regulatory Networks Let the set of all genes of an organism be denoted as $TG = \{tg^1, \ldots, tg^I\}$, and let $TF \subseteq TG$ be the subset of genes encoding TFs. The set of regulatory links between TFs and their TGs is $E \subseteq (TF \times TG)$, such that $(tf, tg) \in E$ means that tf regulates the level of expression of tg. Then, a GRN is simply modeled as an oriented graph $G = \langle TG, E \rangle$, its nodes representing the organism's genes, and its edges being the regulatory interaction between TFs and their TGs.

Data-Driven GRN inference Let us define a function $\omega : \mathbb{R}^{I \times J}, TF, TG \rightarrow \mathbb{R}$ that aims at computing a score $\omega(X, tf, tg)$, to quantify the level of dependency between genes tf and tg. Data-Driven GRN inference rely on such a function, to score all possible regulatory links between TFs and TGs (excluding self-loops), i.e., $E^{full} = \{(tf, tg) \in TF \times TG \quad | \quad tf \neq tg\}$. Finally, a subset of E^{full} is often selected as the inferred GRN, by extracting the links with a score above a given threshold, or selecting the top-k links.

Ensemble of GRN inference methods Let us consider a set of M GRN inference methods $\{m_1, m_2, \ldots, m_M\}$, and let ω_m denote the scoring function of method m. Then, $\Omega = \{\omega_{m_1}, \omega_{m_2}, \ldots, \omega_{m_M}\}$ represents the set of scoring

functions of methods in $\mathcal{M}$. Moreover, let $V : \mathbb{R}^M \rightarrow \mathbb{R}$ be an integration function, that receives as inputs the scores $\omega_m(X, tf, tg)$, $\forall m \in \mathcal{M}$, and outputs a unique final score, that quantifies the consensus level of dependency between tf and tg, for a dataset X. Therefore, an ensemble of GRN inference methods is defined as a pair $\langle \Omega, V \rangle$, containing a set of base scoring functions Ω, and an integration function V.

3.3 Preprocessing

Applying standardisation techniques is an important preliminary step in gene expression data analysis, as in many machine learning tasks [18]. In this work we applied the well-known *Z-score rows* standardization, that ensures that the levels of expression of the different genes are comparable. More formally, each entry $X_{i,j}$ of the gene expression matrix is replaced by $\frac{X_{i,j}-\mu_i}{\sigma_i}$, where $\mu_i = \frac{1}{J}\sum_j X_{i,j}$ is the average gene expression of *gene* i and $\sigma_i = \sqrt{\frac{1}{(J-1)}\sum_j (X_{i,j}-\mu_i)^2}$ represents its standard deviation.

As suggested in [8], the continuous expression vector of each TG was discretized into K levels (classes), using the *Row-Kmeans* method. This method aims at applying the well-known K-means algorithm [19] to cluster the expression values of the TG into K groups. Then cluster memberships are used as discrete gene expressions. More formally, $\forall j \in \{1, \ldots, J\}$ the gene expression values $X_{i,j}$ of *gene* i, are clustered in K clusters, hence C_k denotes the k-th cluster, μ_k is its centroid, and cluster indexes are set according to the centroid location, i.e., $\mu_1 < \mu_2, < \cdots < \mu_K$. Finally, if $X_{i,j} \in C_k$ then $X_{i,j}$ is discretized by taking its cluster index k. As in [8], the number of classes was set to $k = 5$. This value was determined in [8], for the DREAM5 datasets, by identifying the elbow in a plot representing, for different number of clusters, the sum of squared euclidean distance between each gene's expression vector and its cluster centroid.

In practice, the z-score and Row-Kmeans implementations from the GReNaDIne [20] Python package were used.

3.4 Ensembles of GRN inference methods

3.4.1 Base learners training

In order to study ensembles of GRN inference methods, we relied on the GReNaDIne [20] open source Python library, that implements many data-driven gene regulatory network inference methods, that were used here as base learners. In practice, GReNaDIne implements 4 methods based on classical statistical measures, namely Pearson (Pcorr) and Spearman (Scorr) correlations, Kendall-tau (Ktau) and Mutual Information score (MI). Moreover, this package incorporates two methods based on Support Vector Machines (one based on classifiers and one on regressors), and eight methods based on AdaBoost (AB), Gradient Boosting (GB), Random Forest (RF) and eXtreme Randomized Trees (XRT), for both classifiers (c) and regressors (r). This package also includes an implementation

of TIGRESS [17] as well as another similar method based on stability randomized lasso (SRLr). Finally, GReNaDIne includes a method based on Bayesian Ridge Regression (BRr).

The parameters of these methods were set as in [8,20], to the default values, that leaded to suitable results. Similarly, for all the algorithms based on decision trees (i.e. ABc, ABr, GBc, GBr, RFc, RFr, XRTc, XRTr), the number of base estimators, a major parameter, was set to 100 trees. This value ensured a good trade-off between quality, and the execution time (both measures tend to increase with the number of predictors).

3.4.2 Integration scheme

In this work, we used a rather simple integration scheme: first we made the scores distributions comparable between methods, by standardizing them using a z-score, and then we derived the final scores, by averaging the base predictors' standardized scores. This integration scheme does not require to recompute base learners' inferences, and thus it is possible to compute base learners' inferences only once to test many combinations of base methods, saving important computational resources. Notice that other integration schemes, such as the Ranked voting procedure used in [6], worth to be assessed in future works.

More formally, let $S_m = \{\omega_m(X, tf, tg),\ \forall (tf, tg) \in E^{full}\}$, be the set of scores of all possible regulatory links between TFs and TGs, assigned by method $m \in \mathcal{M}$, $\mathcal{M}$ denoting the set of base methods. Moreover, let $\mu_{S_m} = \frac{\sum_{(tf,tg)\in E^{full}} \omega_m(X,tf,tg)}{|E^{full}|}$ and $\sigma_{S_m} = \sqrt{\frac{\sum_{(tf,tg)\in E^{full}} (\omega_m(X,tf,tg)-\mu_{S_m})^2}{|E^{full}|-1}}$ be the average and the standard deviation of scores in S_m respectively. Then, for each regulatory link $(tf, tg) \in E^{full}$, the ensemble score is simply the average of standardized base method scores, i.e., $\omega_{\mathcal{M}}(X, tf, tg) = 1/|\mathcal{M}| \times \sum_{m\in\mathcal{M}} \frac{\omega_m(X,tf,tg)-\mu_{S_m}}{\sigma_{S_m}}$

3.5 Evolution of ensemble candidates

In order to explore the space of combinations of base methods, and select a set of efficient ensembles, we relied in this work on a genetic algorithm. The overall idea is to evolve a population of candidate ensembles, i.e. subsets of the available GRN inference methods, in order to maximize their fitness, i.e., the quality of their inferred GRNs.

The Genetic Algorithm evolves a population of $SizePop$ individuals. Each individual genome encodes an ensemble candidate, and it is represented as a boolean vector with a size equal to the number of the available GRN inference methods (here the 17 GReNaDIne methods presented in Section 3.4.1 are considered). Then, the i-th element of the boolean vector encodes the presence of the i-th base method, in the corresponding ensemble candidate. More formally, let $\mathcal{L} = (m_1, m_2, \ldots, m_L)$ be an arbitrarily ordered list of L methods, and let $B = (b_1, b_2, \ldots, b_L) \quad | \quad \forall b \in B,\ b \in \{0, 1\}$, be a boolean vector of size L. The candidate model encoded by vector B contains a set of methods $\mathcal{M} = \{m \in \mathcal{L}\}$, such that $m_i \in \mathcal{M}$ only if $b_i = 1$. All individuals of a population are initialized

randomly by setting each element to 1 with a probability p_{init} (the higher p_{init} is, the more methods are integrated in the first generation). At each generation, children may mutate with a probability $p_{IndivMut}$. Here, a mutation simply picks randomly one element of the boolean vector with a probability $p_{GeneMut}$, and flips it. And during reproduction, two individuals can undergo a classic two points crossing-over operation with a probability p_{cross}. The parents of the new generation are selected, according to their fitness, using a tournament selection scheme, i.e., $TournSize$ individuals are randomly picked to compete, and the best one is selected to produce $TournSize$ children. The fitness of an individual is computed by evaluating its inferred GRN with respect to a gold standard GRN. More precisely, the fitness of the individual is simply the AUROC evaluation score of the inferred GRN. The AUROC is computed with the procedure described in Section 4.2. Finally the algorithm iterates a mutation step and a selection step, for a number $NbGenerations$ of iterations.

In practice, the genetic algorithm was programmed using the *DEAP* Python library [21], and the meta-parameters were set as follows. The population size was set to $SizePop = 100$ individuals. The mutation probability was set to $p_{IndivMut} = 0.1$ (10% of the population), and $p_{GeneMut} = 1/|B|$ (in average 1 gene is affected). The cross-over probability was set to $p_{cross} = 0.5$, so each new child has a probability of 0.5 to undergo a cross-over. The number of individuals involved in a tournament was set to $TournSize = 5$. Finally, six values have been explored for p_{init}, namely 0.1, 0.2, 0.3, 0.4, 0.5, 0.6 as starting points for the algorithm. Then for each dataset, and each value of p_{init}, 10 populations were evolved independently. For each run, the best individual of the last generation, supposedly the best of all, was kept as a suitable ensemble candidate. Evolving the population for $NbGenerations = 10$ revealed to be sufficient to reach high scores w.r.t. those obtained by base learners, as shown Figure 1. Since our goal is simply to explore the space of promising ensemble candidates, to subsequently mine frequent associations of methods, instead of retrieving the optimal ensemble per dataset, optimizing the meta-parameters was not necessary in this work.

3.6 GRN inference association rules

In order to detect sets of methods that are frequently selected together to form suitable ensemble candidates, we used a frequent itemset mining procedure. In this context, each base method m is considered as an item, and a candidate ensemble comprised of a subset of methods $\mathcal{M}$, is an itemset or transaction. Then, the set of candidate ensembles is a transactions dataset $\mathcal{T} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_T\}$ The support $supp(\mathcal{M}, \mathcal{T})$ of an itemset $\mathcal{M}$ in $\mathcal{T}$, is simply the frequency of itemsets in $\mathcal{T}$ that are supersets of $\mathcal{M}$, i.e., $supp(\mathcal{M}, \mathcal{T}) = \frac{|\{\mathcal{M}_t \in \mathcal{T} \mid \mathcal{M} \subseteq \mathcal{M}_t\}|}{|\mathcal{T}|}$. A transaction $\mathcal{M}$ is called a frequent itemset if $supp(\mathcal{M}, \mathcal{T}) > MinSupp$, where $MinSupp$ is a threshold defined beforehand. Moreover, a frequent itemset $\mathcal{M}$ is said to be maximal if there is no frequent itemset that is a superset of $\mathcal{M}$, i.e., $\nexists \mathcal{M}' \in \mathcal{T} \quad | \quad \mathcal{M} \subset \mathcal{M}'$ and $supp(\mathcal{M}', \mathcal{T}) > MinSupp$.

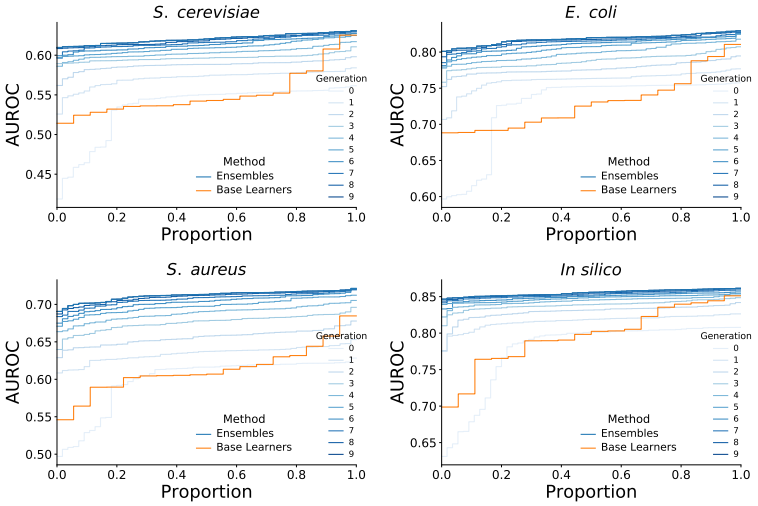


Figure 1: Empirical cumulative distributions for the last populations average scores (blue) and the base-learners ones (orange), for each DREAM5 dataset

With this aim, we used the FP-max [22], a variant of the popular FP-growth algorithm [23], to extract maximal frequent itemsets, from the ensemble candidates dataset. In order, to extract only the most interesting combinations, we set the minimal support threshold to $MinSupp = 0.2$ (i.e., one out of five ensemble candidates should incorporate the itemset). In practice, we used the MLxtend [24] implementation of the FP-max algorithm.

4 Experimental Setup

4.1 Datasets

In order to investigate and assess the use of ensembles of GRN inference methods, we relied on the DREAM5 benchmark data [6]. This benchmark contains three datasets obtained from real organisms, namely *E. coli*, *S. aureus* and *S. cerevisiae*, and an *In silico* simulated dataset. Each dataset is comprised of a gene expression matrix, a list of TFs, and a gold standard GRN, i.e., a list of known regulatory links between TFs and their TGs. Important characteristics of these datasets are reported in table 1.

The gene expression matrices for *E. coli*, *S. aureus* and *S. cerevisiae*, are Affymetrix Microarray datasets, downloaded from Gene Expression Omnibus (GEO)[2] platform. According to [6], these datasets underwent a normalization and filtering procedure that includes: Robust Multichip Averaging background adjustment, quantile normalization, probeset median polishing and logarithmic

[2]http://www.ncbi.nlm.nih.gov/geo

transformation.

In order to determine the TFs lists for *E. coli*, *S. aureus* and *S. cerevisiae*, the authors of [6] conducted Gene Ontology (GO) annotation analysis [6]. Then, they completed *E. coli* and *S. cerevisiae* lists, considering respectively a manually curated TFs list included in the RegulonDB 6.8 database [25] for *E. coli*, and a list of TFs provided in [26] for *S. cerevisiae*.

The gold standard *E. coli* GRN, includes only regulatory links with strong experimental evidence, from the RegulonDB 6.8 database [25]. The gold standard *S. cerevisiae* GRN, includes regulatory interactions that were determined in [27], through the study of ChIP-chip datasets and the query for conserved TF binding sites motifs. Regarding *S. aureus*, the authors of [6] included the prokaryotic regulatory interactions reported in the RegPrecise database [28], as a proxy of a gold standard GRN, since no experimentally validated GRN was available for this organism.

Unlike the previous datasets, the *in silico* dataset, was generated using the GeneNetWeaver software [29]. According to [6], the *In silico* GRN structure is a randomized version of the RegulonDB *E. coli* GRN, that includes 10% of new random regulatory links. This GRN, was used to generate a gene expression matrix, using a dynamical system of Ordinary Differential Equations, based on a multiplicative regulatory interactions model.

Table 1: Benchmark datasets summary

Dataset	Data	# cond.	# genes	# TFs	# Links	$\frac{E_{gold}}{E_{gold}^{full}}$
In silico	Simulated	805	1,643	195	4,012	0.014
S. aureus	Microarray	160	2,810	99	515	0.028
E. coli	Microarray	805	4,511	334	2,066	0.013
S. cerevisiae	Microarray	536	5,950	333	3,940	0.017

4.2 Evaluation

General procedure The evaluation of the GRN inference methods, against gold standards, was conducted following the procedure described in [6] In this procedure, GRN inference is assessed as a binary classification task, where possible regulatory links are classified as true of false. All the links reported in the gold standards, are taken as *true interactions*, for the binary classification. Nevertheless, all the links missing from the gold standards, should be considered as false interactions. Indeed, according to [6], an organism's GRN gold standard only contains the *experimentally tested subset* of all its true regulatory interactions. Therefore, in order to avoid penalizing methods for detecting true interactions remaining experimentally untested, any link involving a TF or a TG that was not studied experimentally is excluded from the assessment [6]. Only pairs missing from the gold standard list, and involving both a TF and a TG experimentally studied, are taken as *false interactions*.

Formal definition Let TG and $TF \subset TG$ be respectively a set of genes and the subset of genes encoding TFs. Let the oriented graph $G_{gold} = \langle TF_{gold} \cup TG_{gold}, E_{gold} \rangle$ be a gold standard GRN, with $TF_{gold} \subseteq TF$ and $TG_{gold} \subseteq TG$ being respectively the set of experimentally studied TFs and TGs, and $E_{gold} \subseteq E^{full}_{gold}$ being the set of *true regulatory links* among the set of possible links $E^{full}_{gold} = TF_{gold} \times TG_{gold}$. Links in $E^{full}_{gold} \setminus E_{gold}$ are considered as *false regulatory links*, while links in $(TF \times TG) \setminus E^{full}_{gold}$ are not taken into account in the evaluation. The fraction of true regulatory links E_{gold}/E^{full}_{gold}, reported in Table 1 shows that the datasets exhibit a strong class imbalance.

Evaluation measures As in [6] we assessed the methods using standard evaluation measures for binary classification, from the machine learning community, namely the Area Under the Receiver Operating Characteristic curve (AUROC) [30], and the Area Under the Precision Recall curve (AUPR) [31] values.

4.3 Experimental protocol

Comparison with DREAM5 ensemble In order to assess the ensemble candidates, studied and proposed in this paper, we compared their AUROC and AUPR scores, with respect to those obtained by the ensemble of DREAM5 participants. The performance measures obtained by the DREAM5 ensemble, on each benchmark dataset, as defined in Section 4.2, have been made available by [6]. The single GRN inference methods implemented in GReNaDIne [20], as well as the ensemble candidates presented here, were executed on the DREAM5 benchmark datasets, and their results were assessed against the gold standards networks, following the procedure described in Section 4.2.

Base learners diversity exploration According to [1, 6], the performance and the robustness of ensembles increase, when the base learners are diverse. In order to study the similarities between GReNaDIne predictors, we have selected E^{top}, the regulatory interactions that were among the top 50,000 links of at least one base predictor in one dataset, yielding a total of $|E^{top}| = 419,904$ links, from the different datasets. Then each link was represented in the base predictors rank space: let R^{top} be a matrix with $|\mathcal{M}|$ columns, and $|E_{top}|$ rows, such that element $R^{top}_{i,j}$ denotes the ranking of the score assigned by method j to the link i (rank 1 being assigned to the highest score). Finally, the R^{top} matrix was standardized using a column z-score, and then we applied the principal component analysis [32], to represent the methods in the two first Principal Components (PC) space.

All experiments were executed on a Intel(R) Xeon(R) 2.40GHz CPU, running Debian GNU/Linux 10, with a 120 Go RAM capacity.

5 Results

Following the aforementioned experimental protocol, seven maximal frequent itemsets, denoting suitable combinations of base methods, were detected among the best ensemble candidates. Three out of these combinations, denoted BRSr•SVMr•Trees, are composed of BRSr, SVMr and a tree-based approach (i.e., RFc, XRTc, ABr), three other maximal frequent itemsets are combinations of tree-based methods (i.e., ABR•GBr, GBc•XRTr and ABr•GBc), and the last one is the combination of BRSr and SVMc.

Combinations of tree-based methods were mostly selected in the *In silico* dataset, while other combinations were selected in real datasets, as show the itemset supports per dataset depicted in Figure 2. Moreover, tree-based ensembles exhibit better AUROC and AUPR scores, for the *In silico* dataset, and mediocre results for the real datasets, while the remaining combinations exhibit decent results for the *In silico* dataset, and among the best results for the real datasets, as shown in Figure 2. BRSr•SVMr•Trees revealed to be the most interesting combination, that dominates the individual methods, as well as the other combinations, as depicted Figure 3. Thus, even small ensembles of three methods may be sufficient to have efficient and robust performances. Regarding evaluation scores, all the methods, including single ones, exhibit lower AUPR than AUROC scores, as shown Figure 3. This may be due to the datasets class imbalance, since in this context AUCROC is less sensitive than AUPR [31].

In order to understand the efficiency of the BRSr•SVMr•Tree ensembles, we investigated the relatedness between base learners, using the Principal Component Analysis, as described in the previous section. As shown in Figure 4, the two highest principal components reveal clusters of methods based on: 1) MI or correlation measures, 2) ensembles of trees, or ensembles of regularized linear regressors 3) SVMs 4) BRSr as an outlier. The methods belonging to the same cluster, are likely to share the same intrinsic biases [6]. Thus including one method from each cluster, is likely to produce an ensemble with a high inner diversity, that would compensate the base-learners biases. And this is likely to be the reason behind the efficiency of BRSr•SVMr•Tree ensembles.

In order to assess this hypothesis, we computed the AUROC and AUPR scores for ensemble containing BRSr, an SVM-based method, and an ensemble based method (termed BRSr•SVM•Ens), as well as for ensembles containing one base-learner from each of cluster of methods. As shown Figure 5, both families of ensembles exhibit significantly better results than base-learners and than the DREAM5 ensemble for the real datasets, and comparable results for the *In silico* dataset. Therefore, the efficiency of the BRSr•SVMr•Tree ensembles, detected using the itemset mining technique, seems to be explained by this general principle. Furthermore, including one extra method from the correlation-based family, did not improve the results, for these datasets, but could be beneficial to deal with other datasets.

Finally, in order to compare the SVM•BRSr•Tree methods, we represented in Figure 6, the ranking of each combination regarding its AUROC and AUPR score on each dataset, rank 1 being assigned to the best method, and rank

16 to the worst one. According to these results, the most suitable and robust ensembles, combine BRSr, SVMr and one of the following tree-based methods: RFr, RFc, XRTc or ABr.

6 Conclusion

This paper explored the use of ensemble learning, as a robust and efficient approach to infer GRNs, from gene expression data. In practice, ensemble predictions were computed by averaging the results from single GRN inference methods, implemented in the GReNaDIne framework [20]. Efficient combinations of methods were designed using a genetic algorithm and a frequent itemset mining procedure. The resulting ensembles, termed BRSr•SVMr•Tree, revealed to be efficient and robust across different datasets, outperforming single methods as well as the robust community method presented in [6]. A subsequent analysis revealed that the effectiveness of BRSr•SVMr•Tree is due to the inner diversity of its base learners. This result is coherent, with a well-known ensemble learning principle, that affirms that increasing the diversity of base methods, tends to improve the ensemble quality, through the compensation of base-learners biases [1], and a better exploration of the solution space, that is extended beyond the base learners' solution spaces.

Future work perspectives include studying: i) more sophisticated combination schemes ii) internal single methods biases, to retrieve specific patterns, and how can ensembles reduce the biases iii) strategies to deal with class imbalance iv) genetic algorithm and frequent itemset parameters sensitivity.

References

[1] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.

[2] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, pp. 1–18, 2019.

[3] P. Yang, Y. Hwa Yang, B. B Zhou, and A. Y Zomaya, "A review of ensemble methods in bioinformatics," *Current Bioinformatics*, vol. 5, no. 4, pp. 296–308, 2010.

[4] G. Sanguinetti and V. A. Huynh-Thu, "Gene regulatory network inference: an introductory survey," in *Gene Regulatory Networks*. Springer, 2019, pp. 1–23.

[5] D. Latchman, *Gene regulation*. Taylor & Francis, 2007.

[6] D. Marbach, J. C. Costello, R. Küffner, N. M. Vega, R. J. Prill, D. M. Camacho, K. R. Allison, D. Consortium, M. Kellis, J. J. Collins, and

G. Stolovitzky, "Wisdom of crowds for robust gene network inference," *Nature methods*, vol. 9, no. 8, p. 796, 2012.

[7] L. E. Chai, S. K. Loh, S. T. Low, M. S. Mohamad, S. Deris, and Z. Zakaria, "A review on the computational approaches for gene regulatory network construction," *Computers in biology and medicine*, vol. 48, pp. 55–65, 2014.

[8] S. Peignier, P. Schmitt, and F. Calevro, "Data-driven gene regulatory network inference based on classification algorithms," in *2019 IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2019, pp. 1–8.

[9] S. Aibar, C. B. González-Blas, T. Moerman, H. Imrichova, G. Hulselmans, F. Rambow, J.-C. Marine, P. Geurts, J. Aerts, J. van den Oord *et al.*, "Scenic: single-cell regulatory network inference and clustering," *Nature methods*, vol. 14, no. 11, p. 1083, 2017.

[10] A. Irrthum, L. Wehenkel, and P. Geurts, "Inferring regulatory networks from expression data using tree-based methods," *PloS one*, vol. 5, no. 9, p. e12776, 2010.

[11] W. Zhuang, Y. Ye, Y. Chen, and T. Li, "Ensemble clustering for internet security applications," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1784–1796, 2012.

[12] T. Moerman, S. Aibar Santos, C. Bravo González-Blas, J. Simm, Y. Moreau, J. Aerts, and S. Aerts, "Grnboost2 and arboreto: efficient and scalable inference of gene regulatory networks," *Bioinformatics*, vol. 35, no. 12, pp. 2159–2161, 2019.

[13] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[14] J. H. Friedman, "Stochastic gradient boosting," *Computational statistics & data analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[15] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine learning*, vol. 63, no. 1, pp. 3–42, 2006.

[16] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," *Journal-Japanese Society For Artificial Intelligence*, vol. 14, no. 771-780, p. 1612, 1999.

[17] A.-C. Haury, F. Mordelet, P. Vera-Licona, and J.-P. Vert, "Tigress: trustful inference of gene regulation using stability selection," *BMC systems biology*, vol. 6, no. 1, p. 145, 2012.

[18] C. Cheadle, M. P. Vawter, W. J. Freed, and K. G. Becker, "Analysis of microarray data using z score transformation," *The Journal of molecular diagnostics*, vol. 5, no. 2, pp. 73–81, 2003.

[19] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[20] S. Peignier, P. Schmitt, and F. Calevro, "Grenadine: data-driven approaches to infer gene regulatory networks in python," Jun. 2020, working paper or preprint. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02863880

[21] F.-M. De Rainville, F.-A. Fortin, M.-A. Gardner, M. Parizeau, and C. Gagné, "Deap: A python framework for evolutionary algorithms," in *Proceedings of the 14th annual conference companion on Genetic and evolutionary computation*, 2012, pp. 85–92.

[22] G. Grahne and J. Zhu, "Efficiently using prefix-trees in mining frequent itemsets." in *FIMI*, vol. 90, 2003, p. 65.

[23] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data mining and knowledge discovery*, vol. 8, no. 1, pp. 53–87, 2004.

[24] S. Raschka, "Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack," *The Journal of Open Source Software*, vol. 3, no. 24, Apr. 2018. [Online]. Available: http://joss.theoj.org/papers/10.21105/joss.00638

[25] S. Gama-Castro, H. Salgado, M. Peralta-Gil, A. Santos-Zavaleta, L. Muniz-Rascado, H. Solano-Lira, V. Jimenez-Jacinto, V. Weiss, J. S. Garcia-Sotelo, A. Lopez-Fuentes *et al.*, "Regulondb version 7.0: transcriptional regulation of escherichia coli k-12 integrated within genetic sensory response units (gensor units)," *Nucleic acids research*, vol. 39, no. suppl_1, pp. D98–D105, 2010.

[26] C. Zhu, K. J. Byers, R. P. McCord, Z. Shi, M. F. Berger, D. E. Newburger, K. Saulrieta, Z. Smith, M. V. Shah, M. Radhakrishnan *et al.*, "High-resolution dna-binding specificity analysis of yeast transcription factors," *Genome research*, vol. 19, no. 4, pp. 556–566, 2009.

[27] K. D. MacIsaac, T. Wang, D. B. Gordon, D. K. Gifford, G. D. Stormo, and E. Fraenkel, "An improved map of conserved regulatory sites for saccharomyces cerevisiae," *BMC bioinformatics*, vol. 7, no. 1, p. 113, 2006.

[28] P. S. Novichkov, O. N. Laikova, E. S. Novichkova, M. S. Gelfand, A. P. Arkin, I. Dubchak, and D. A. Rodionov, "Regprecise: a database of curated genomic inferences of transcriptional regulatory interactions in prokaryotes," *Nucleic acids research*, vol. 38, no. suppl_1, pp. D111–D118, 2009.

[29] T. Schaffter, D. Marbach, and D. Floreano, "Genenetweaver: in silico benchmark generation and performance profiling of network inference methods," *Bioinformatics*, vol. 27, no. 16, pp. 2263–2270, 2011.

[30] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[31] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 233–240.

[32] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

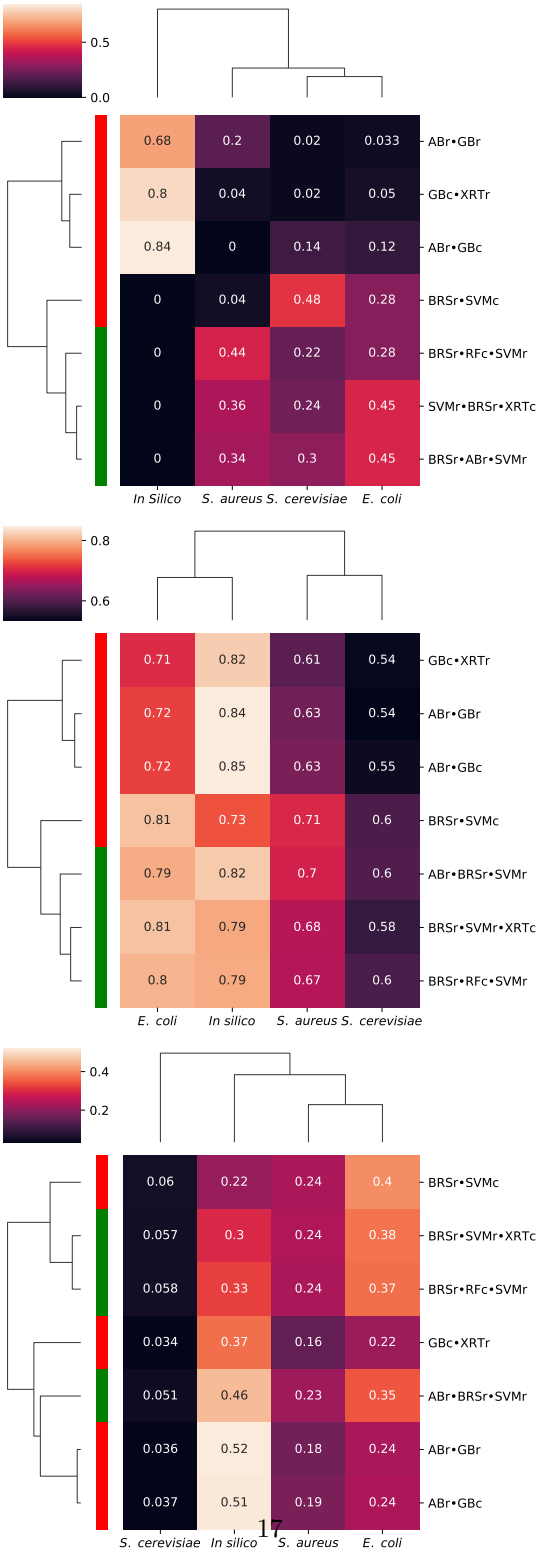


Figure 2: Support, AUROC and AUPR scores for the maximal itemsets extracted, computed for each dataset. Ensembles that include SVMr, BRSc and a method based on decision trees are represented in green, and other combinations in red.

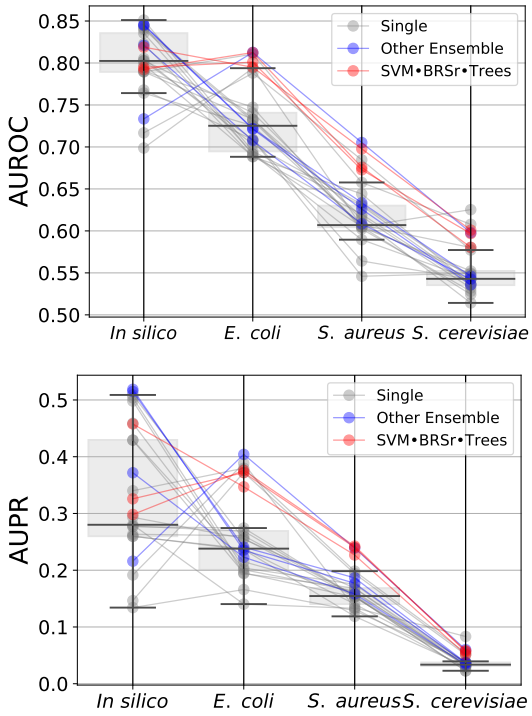


Figure 3: AUROC and the AUPR scores obtained by each GRN inference algorithm from GReNaDIne, on the DREAM5 dataset (gray). Single methods parallel plots and boxplots are represented in gray, while BRSr•VSMr•Trees ensembles parallel plots are depicted in red, and other ensembles in blue.

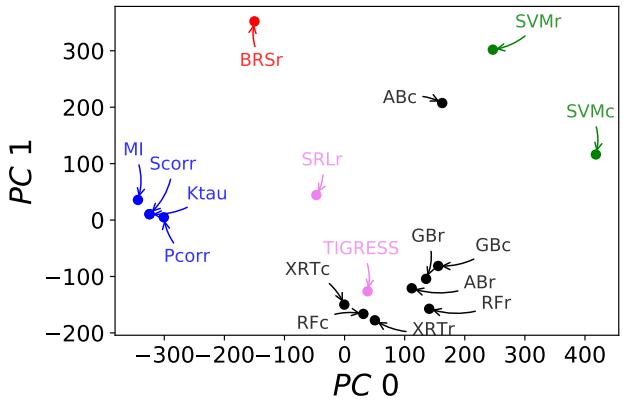


Figure 4: Single methods represented along the first and second Principal components.

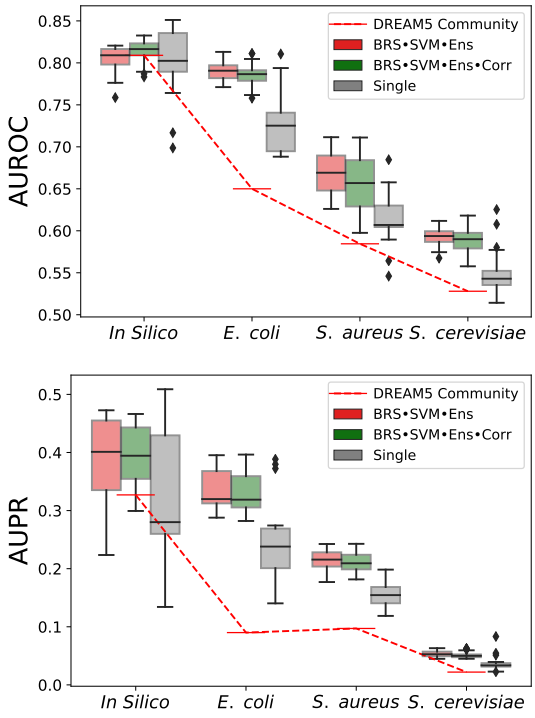


Figure 5: Boxplots representing, for each DREAM5 dataset, the AUROC and the AUPR obtained by SVM•BRS•Ens ensembles (red), SVM•BRS•Ens•Corr ensembles (green), single GReNaDIne methods (gray) and the DREAM5 community (red line)

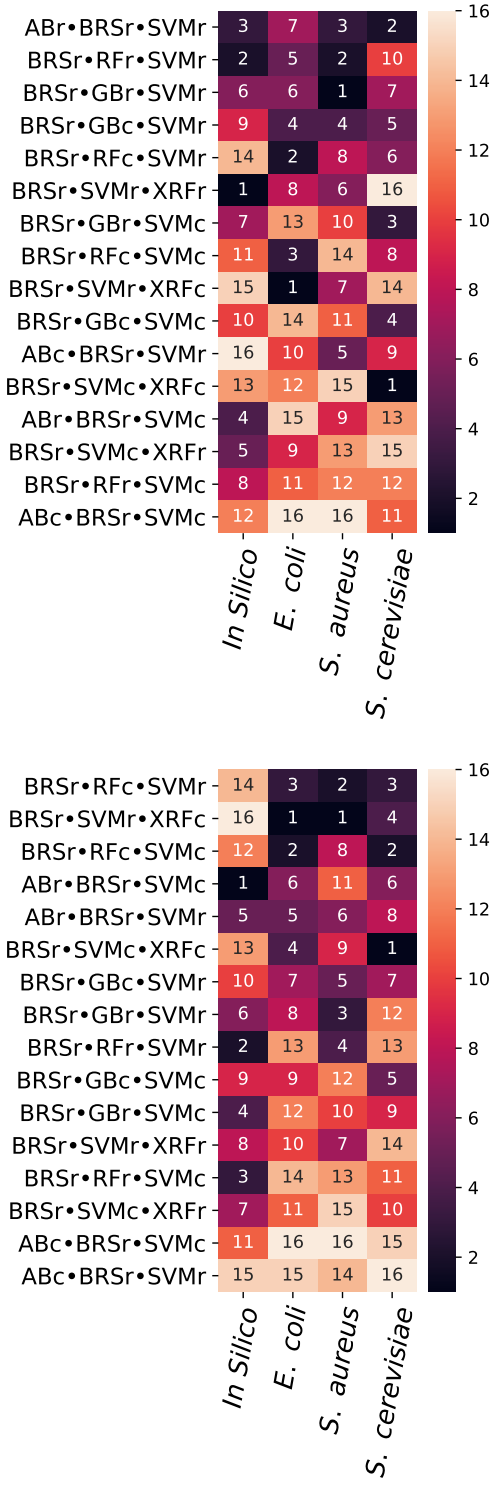


Figure 6: Heatmaps representing, for each dataset, the ranking of SVM•BRSr•Tree combinations, regarding their AUROC (top) and AUPR (bottom) evaluation scores (rank 1 being assigned to the best method)