UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Machine learning tools for bioinformatics problems

Victor Alexandre Padilha

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

Victor Alexandre Padilha

Machine learning tools for bioinformatics problems

Doctoral dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho
Co-advisor: Prof. Dr. Rolf Backofen

USP – São Carlos
November 2020

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

P123m

Padilha, Victor Alexandre
Machine learning tools for bioinformatics problems / Victor Alexandre Padilha; orientador André Carlos Ponce de Leon Ferreira de Carvalho; coorientador Rolf Backofen. -- São Carlos, 2020.
154 p.

Tese (Doutorado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2020.

1. CRISPR-Cas. 2. Expressão Gênica. 3. Bioinformática. 4. Aprendizado de Máquina. 5. Inteligência Artificial. I. de Carvalho, André Carlos Ponce de Leon Ferreira, orient. II. Backofen, Rolf, coorient. III. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Victor Alexandre Padilha

Ferramentas de aprendizado de máquina para problemas de bioinformática

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho
Coorientador: Prof. Dr. Rolf Backofen

USP – São Carlos
Novembro de 2020

To my beloved parents, Osmário and Walkiria.

ACKNOWLEDGEMENTS

This thesis is the outcome of almost four years of hard work. During this time, a lot of people directly or indirectly contributed to it. It is satisfying to look back and see that I was lucky to have many awesome people by my side, with whom I could share some of the best years of my life. I know that I will probably forget someone, but I am thankful for every person that was part of this journey.

First and foremost I would like to thank my family, for always supporting me in my decisions and giving me all the love that I needed to not give up. Especially, I would like to thank my parents, Osmário Padilha Jr. and Walkiria Chriezanoski Padilha.

I thank my girlfriend, Daiane Canhoni, for always being so lovely and patient with me, especially during my hardest moments. For always listening to me whenever I needed and for all the emotional support during these years.

To my advisor, Prof. André C. P. L. F. de Carvalho, for these years of collaboration, friendship and for keeping me motivated, particularly when our research and experiments were not going well. He has been an example for me not only as a professional but also as a person. It was a pleasure to be under his supervision.

To Prof. Maria Emilia M. T. Walter, from the University of Brasilia (UnB), for giving me the opportunity to have one of the most important experiences of my life, which was to spend one year abroad. It was essential for my personal and professional development.

To my co-advisor, Prof. Rolf Backofen, for our research collaboration and for receiving me as a visiting student in his group at the University of Freiburg, and to my friends and colleagues during the year I spent abroad. Especially, I would also like to thank Omer S. Alkhnbashi, Van Dinh Tran and Florian Eggenhofer.

To João Victor Dutra Gomes for the friendship and the time we spent together in Germany, and to Atavi Yenke for helping me with my first steps in that country and for all conversations we had on the bus, either on the way to the university or back home.

To Prof. Sandro Rautenberg and Prof. Fábio Hernandes, from the Central-Western State University of Paraná (UNICENTRO), where I did my undergraduate studies. For all the advices and for motivating me to do my graduate studies at the University of São Paulo.

To Leila Sanches for the therapeutic support, which helped me to know myself better and to reduce my stress and anxiety.

To my friends from my hometown (Irati–PR): Enzo Crisigiovanni, Ciro Bittencourt, Giovany Pauluk, Cristiano Ribas, Romeu Batista Netto, Tiago Batista, Mateus Zanlorenzi, André Vosnika and Guilherme Kosinski.

To my friends from my undergraduate years in Guarapuava–PR: Guilherme Mello, Guilherme Leão, Leandro Loma, Luis Simões, Gabriel Cecchin, Willian Yassue and Alexandre Antoniu Neto.

To my friends from São Carlos–SP, which were my family during the last few years as a graduate student. For all the moments we shared together. I would like to thank Leandro Mundim, Arianne Alves, Luiz Cherri, Paulo Fontoura Jr., Wilker Fernandes, Adam Henrique Moreira, Rafael G. Mantovani, Daniel Cestari, Murillo Batista, Victor Barella, Henrique Oliveira, Luis Paulo Garcia, Daniel Tozadore, Caetano Ranieri, Edesio Alcobaça, Guilherme Nardari, Adriano Rivolli, Saulo Mastelini, Gean Trindade, Everlandio Fernandes, Jefferson Oliva, Moisés Rocha, Jadson Castro, Valéria Carvalho, José Pedro Belo, Raphael Montanari, Marcelo Silva, William Rosa, Lucas Pagliosa, Daniela Ridel, Valdemar Devesse, Arthur Fortes, Francisco Carlos Monteiro, Alinne Corrêa, Wilmax Cruz, Gisele Aguilar, Lívia Degrossi, Diógenes Dias, Misael Costa Jr., João Paulo Moreira, Stevão Andrade, Faimison Porto, Carlos Diego Damasceno, Brauner Oliveira, Danillo Reis, Edvaldo Neto and Geovanna Batista.

Last but not least, I would like to thank the São Paulo Research Foundation (FAPESP) (grants #2017/02975-0 and #2019/21300-9) and the Federal Agency for Support and Evaluation of Graduate Education within the Ministry of Education of Brazil (CAPES) (Finance Code 001 and Probral CAPES/DAAD grant #88887.302257/2018-00) for the financial support.

"Science is the great antidote to the poison of enthusiasm and superstition."

Adam Smith

An Inquiry Into the Nature and Causes of the Wealth of Nations

ABSTRACT

PADILHA, V. A. **Machine learning tools for bioinformatics problems**. 2020. 154 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2020.

In recent years, machine learning techniques have been extensively used for bioinformatics, due to their capacity in solving hard problems by learning a function from a set of known examples, being this function able to make predictions for unseen data. Motivated by these successful applications, we tackle in this thesis three different bioinformatics problems using machine learning techniques.

The first problem is related to the use of coherence measures for the analysis of biclustering results in gene expression data analysis. Specifically, we conducted a detailed investigation of the correlations between different bicluster coherence measures on a benchmark of 19 datasets of the *Saccharomyces cerevisiae* organism. We were able to identify pairs of redundant measures and also observed that such measures did not present any relation with external knowledge available in the form of gene ontologies.

The second problem is related to the classification of CRISPR cassettes into their subtypes and the prediction of potentially missing proteins. We proposed a novel tool, called CRISPRcasIdentifier, which integrates classifiers and regressors for these tasks. It outperformed the competitors from the literature on the most recent benchmark dataset available and is the first tool that is able to recommend potentially missing proteins in CRISPR cassettes.

The third problem is related to the automatic identification of CRISPR cassettes in bacterial and archaeal genomes. We introduced Casboundary, a new tool that detects CRISPR cassettes based on gene signatures and their relations with neighboring genes. Moreover, this tool is able to point out potentially new *cas* genes, as demonstrated by a case study. Finally, Casboundary is also capable of decomposing a CRISPR cassette into its modules, which are related to the different stages of the CRISPR systems.

Keywords: Machine Learning, Biclustering, Gene Expression Data Analysis, CRISPR-Cas Systems, Cas Proteins.

RESUMO

PADILHA, V. A. **Ferramentas de aprendizado de máquina para problemas de bioinformática**. 2020. 154 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2020.

Recentemente, técnicas de aprendizado de máquina têm sido utilizadas de maneira extensiva em problemas de bioinformática, devido à sua capacidade na resolução de problemas complexos por meio do aprendizado de uma função a partir de uma amostra finita de exemplos, sendo tal função capaz de realizar predições para novos dados. Motivado por essas aplicações bem sucedidas, este trabalho aborda três problemas diferentes de bioinformática por meio de técnicas de aprendizado de máquina.

O primeiro problema está relacionado ao uso de medidas de coerência para a análise de resultados de bi-agrupamento em análise de dados de expressão gênica. Especificamente, foi conduzida uma investigação detalhada acerca das correlações entre diferentes medidas de coerência de bi-grupos em uma coleção de 19 bases de dados do organismo *Saccharomyces cerevisiae*. Com isso, tornou-se possível identificar pares de medidas redundantes e observar que tais medidas não apresentam qualquer relação com conhecimento externo disponível no formato de ontologias de genes.

O segundo problema está relacionado à classificação de instâncias do sistema CRISPR em seus diferentes subtipos e a predição de proteínas potencialmente ausentes em tais instâncias. Para isso, uma nova ferramenta, chamada CRISPRcasIdentifier, foi proposta, a qual integra modelos de classificação e regressão para as tarefas mencionadas. Tal ferramenta atingiu melhores resultados do que os competidores encontrados na literatura na base de dados mais recente disponível. Ademais, a CRISPRcasIdentifier é a primeira ferramenta capaz de recomendar proteínas potencialmente ausentes em instâncias do sistema CRISPR.

O terceiro problema está relacionado à identificação automática de instâncias do sistema CRISPR em genomas de organismos bacterianos e archaeanos. Para isso, a ferramenta Casboundary foi proposta, a qual detecta instâncias do CRISPR ao considerar as relações entre genes assinatura com seus vizinhos. Além disso, esta ferramenta é capaz de apontar genes *cas* potencialmente novos, tal como demonstrado em um estudo de caso. Finalmente, a ferramenta Casboundary é capaz de decompor as instâncias do CRISPR em seus diferentes módulos, os quais estão relacionados aos diferentes estágios do sistema CRISPR.

Palavras-chave: Aprendizado de Máquina, Bi-Agrupamento, Análise de Dados de Expressão Gênica, Sistemas CRISPR-Cas, Proteínas Cas.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

CHAPTER

1

INTRODUCTION

Machine Learning (ML) research investigates the development of new algorithms able to learn specific tasks by identifying patterns given a finite sample of data. Research in this area dates back to pioneering studies of Artificial intelligence in the 1950s, such as the famous Turing Test (TURING, 1950) and the proposal of the Perceptron neural network (ROSENBLATT, 1957). In the past few decades, it has attracted a large deal of attention, given the large amounts of data generated for several applications every day and the increasing processing capabilities of computers, which facilitate the process of building models that are suitable for complex real-world problems. One of the areas that has benefited from the use of ML algorithms is the Bioinformatics.

Bioinformatics can be described by its three main objectives (LUSCOMBE; GREENBAUM; GERSTEIN, 2001): (i) to organize large amounts of unstructured biological data into structured formats that can be easily retrieved and updated by specialists; (ii) to develop tools that speed up the analysis of biological data by efficiently executing pipelines on structured data; and (iii) to help in the extraction of meaningful biological knowledge, by identifying patterns and novelties in these data. In this thesis, we proposed and experimentally investigated ML approaches and tools to be applied to gene expression data analysis and classification of Clustered Regularly Interspaced Short Palindromic Repeats (CRISPR) systems.

This chapter is organized as follows. Section 1.1 introduces the biological concepts and tools that were explored by us in the gene expression data analysis problem. Section 1.2 provides an overview of CRISPR systems, which have attracted the interest of the Bioinformatics community due to their high potential for genome engineering and gene editing applications. Section 1.3 summarizes our main contributions from this thesis. Finally, Section 1.4 presents an overview of the chapters that compose this thesis.

1.1 Gene expression data analysis

In this section, we introduce fundamental biological concepts of gene expression analysis necessary to understand some of the problems approached by this thesis.

1.1.1 Biological concepts

Proteins are large molecules composed by smaller molecules called amino acids, which are organized in a chain format[1]. They are responsible for many important functions in living organisms, such as: structural proteins that constitute tissues and cells, enzymes that speed up biochemical reactions, oxygen transport, antibody defense, etc (SETUBAL; MEIDANIS, 1997). All the information necessary to build proteins is stored in the deoxyribonucleic acid (DNA). During the gene expression, this information is copied into a ribonucleic acid (RNA) molecule and carried to a sophisticated machinery called the ribosome, which is responsible to assemble the protein. In this section, we give a brief overview about this process. For a more detailed and technical view, we recommend the book of Lodish *et al.* (2008).

The DNA encodes the genetic information and is composed by simpler molecules called nucleotides, which are identified by their base pairs. There are four base pairs that constitute the DNA: adenine (A), guanine (G), cytosine (C) and thymine (T). The DNA is found in a double helix structure, where the nucleotides from one helix are paired by hydrogen bonds to the nucleotides from the other helix following two rules: A pairs with T and G pairs with C (WATERMAN, 1995). An example of the DNA structure is shown in Figure 1a. Inside the DNA there are some contiguous regions, known as genes, that store the information to build specific proteins. The genes are identified by sequences called promoters, which indicate to the cellular mechanism the position where a gene or group of genes starts in the DNA (SETUBAL; MEIDANIS, 1997).

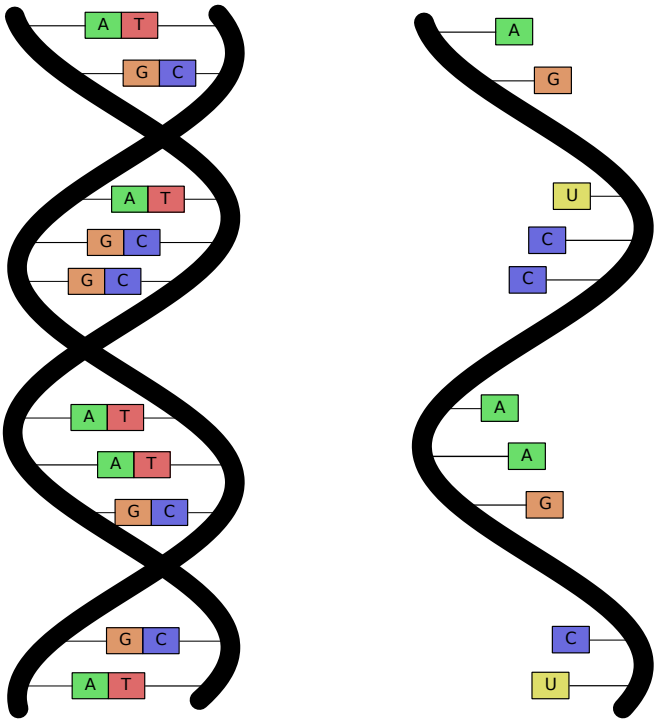
The RNA is usually found as a single helix molecule. Similarly to the DNA, it is also composed by four base pairs: A, C, G and U, where U stands for Uracil. The pairing rules are maintained with U now pairing with A (WATERMAN, 1995; SETUBAL; MEIDANIS, 1997). An example of the RNA structure is illustrated in Figure 1b.

The Central Dogma of Molecular Biology explains how the information that is stored by the DNA is transcribed into an RNA and then translated into a protein (ALKHNBASHI, 2017). It is illustrated in Figure 2 and consists of four different processes (SETUBAL; MEIDANIS, 1997):

1. *Replication*: which is responsible for duplicating the DNA molecule. This process allows an organism to grow from a single cell to billions of cells that are similar to the original

[1] According to Lodish *et al.* (2008), proteins usually range between 100 and 1000 amino acids and a "typical" protein has ~ 400 amino acids.

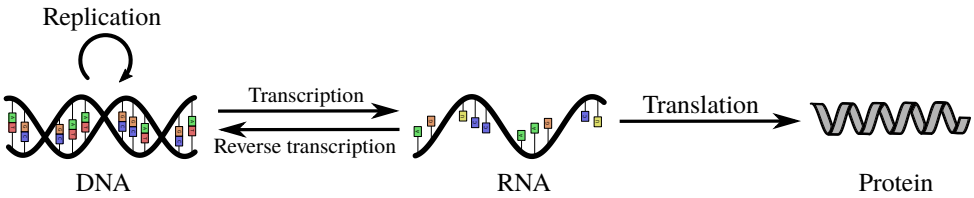
Figure 1 – Illustrations of the DNA and RNA structures.



(a) DNA (b) RNA

Source – Elaborated by the author.

Figure 2 – The Central Dogma of Molecular Biology.



Source – Elaborated by the author.

one.

2. *Transcription*: which copies the information of a gene into a messenger RNA molecule (mRNA). In eukaryotes, which are organisms whose cells contain a nucleus, some regions that are not necessary to build the protein are removed from this molecule. These regions are called introns, while the regions that remain are called exons.

3. *Reverse transcription*: which produces a DNA molecule from an mRNA molecule.

4. *Translation:* which is responsible for translating the information stored in the mRNA into a protein. For such, the mRNA is transported to the ribosome, where it is read in codons. Each codon consists of a nucleotide triplet and refers to a specific amino acid.

Gene expression is the process of using the information stored by a gene to perform the protein synthesis and is directly related to the transcription step described above (LODISH *et al.*, 2008). The levels of expression of multiple genes are measured by experiments with high-throughput technologies among which microarrays became widely used (BROWN; BOTSTEIN, 1999). These technologies quantify the abundance of mRNA of different genes across different samples, which vary from study to study, depending on its objective. One common example is in the study of cancer, where one may collect cancerous tissues from different types and subtypes (RHODES *et al.*, 2004; SOUTO *et al.*, 2008). After the results of multiple experiments are collected, they can be presented in a matrix form:

$$X = \begin{array}{c} \\ \text{Genes} \end{array} \overset{\text{Biological samples}}{\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{12} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}}, \tag{1.1}$$

where x_{ij} is a real value that corresponds to the abundance of mRNA of the i^{th} gene in the j^{th} sample (MADEIRA; OLIVEIRA, 2004).

Since gene expression matrices are composed by a large number of genes and samples, clustering algorithms have been widely used to help researchers to better understand the data to be analyzed by identifying hidden patterns across multiple gene expression data obtained by different technologies, like microarrays (ZHANG, 2006) and RNAseq (WANG; GERSTEIN; SNYDER, 2009).

1.1.2 Data clustering and biclustering

Data clustering is an exploratory data analysis task whose objective is to find a structure that organizes a set of objects $X = \{x_1, \cdots, x_n\}$ into k subsets called clusters (JAIN; DUBES, 1988; TAN; STEINBACH; KUMAR, 2006). For such, data clustering algorithms consider a $n \times m$ data matrix representation of X, where each object is a vector described by m features, and some measure that quantifies the (dis)similarity between pairs of objects considering all features. As a result, it is expected that objects that are clustered together present high similarities (resp. low dissimilarities) to each other and low similarities (resp. high dissimilarities) to the objects of other clusters (TAN; STEINBACH; KUMAR, 2006). Formally, a hard clustering partition $\mathbb{C} = \{C_1, \cdots, C_k\}$ of k clusters is defined by Xu and Wunsch (2005) as:

1. $\bigcup_{i=1}^{k} C_i = C_1 \cup C_2 \cup \cdots \cup C_k = X$;
2. $C_i \neq \emptyset \quad \forall i \in \{1, \cdots, k\}$;
3. $C_i \cap C_j = \emptyset \quad \forall i, j \in \{1, \cdots, k\}$ and $i \neq j$.

Traditional clustering algorithms have been successfully applied to a wide variety of tasks in gene expression data analysis studies, such as: to understand the functions of some genes that were not known beforehand, understand the cellular processes that genes contained in the same cluster take part in, identify potential subcell types, etc (JIANG; TANG; ZHANG, 2004). However, the use of these techniques for gene expression data analysis have two main limitations (MADEIRA; OLIVEIRA, 2004; PADILHA; CAMPELLO, 2017):

1. Frequently, the genes contained in the same cluster present similarity only for subsets of the biological samples, due to the high dimensionality of the data available. In addition, the subset of samples may vary from cluster to cluster.

2. Most traditional clustering techniques produce partitions that are mutually exclusive (i.e., clusters do not share objects) and exhaustive (i.e., all objects are clustered). However, genes and samples may be involved in multiple biological processes and belong to one cluster, multiple clusters[2], or no cluster.

Thus, the biclustering paradigm was proposed to solve these these limitations, by providing algorithms that are able to detect clusters of genes contained in subsets of the available samples (KRIEGEL; KRÖGER; ZIMEK, 2009). The search procedures employed by these algorithms optimize different criteria that are analogous to the notion of (dis)similarity in the traditional clustering literature. Such criteria are tipically called coherence measures.

Biclustering analysis dates back to the 1970s, when Hartigan (1972) proposed the first algorithm able to cluster a data matrix into a set of submatrices. In that paper, the author developed a divide-and-conquer procedure to help in the analysis of Republican voting data of the United States of America, where the rows represented different states and the columns different election years. Afterwards, Cheng and Church (2000) introduced a greedy algorithm focused on gene expression data analysis, which revolutionized the literature. Their work motivated further developments and expanded the applications of biclustering to varied types of problems. However, the main researches are still conducted in the bioinformatics area (MADEIRA; OLIVEIRA, 2004; PONTES; GIRÁLDEZ; AGUILAR-RUIZ, 2015; XIE *et al.*, 2019).

Although many studies have proved the relevance of biclustering techniques for gene expression data analysis, the evaluation of results in a quantitative manner is still a challenging task. The main problem arises from the fact that, to the best of our knowledge, there are no gene expression datasets with labeled biclusters publicly available. To overcome this limitation, some studies focus on the evaluation through bicluster coherence measures. However, it is hard to compare the results of different algorithms using this approach, because they are usually based on different coherence measures. Other studies use domain knowledge, available in the form of gene ontologies. We describe these ontologies in the next section.

[2] In this case, the clusters may be overlapped. The overlap may exist on genes, on samples or on both dimensions simultaneously.

1.1.3 Gene Ontology

According to Ashburner *et al.* (2000), many organisms present similar genes, which motivates the unification of the information available about them, since the knowledge about a specific gene in one organism likely helps to identify/predict the roles that it takes in other organisms. The Gene Ontology (GO) is a collaborative project for the unification of all the knowledge available for different genes in an interpretable structure. This structure consists of a standardized vocabulary, which is organized as a directed acyclic graph, where each node represents a term and each edge refers to the relation between two terms, such as "is-a" and "part-of" (e.g., see Figure 3) (CONSORTIUM, 2004). Each term may refer to one or multiple genes and is labeled with a unique GO identifier. There are three different ontologies available (ASHBURNER *et al.*, 2000; CONSORTIUM, 2004):

- *Cellular Component ontology*: refers to the locations where the material produced by genes are active inside the cells. Examples of terms from this ontology are: "cell body" (GO:0044297), "cytoplasm" (GO:0005737) and "lateral part of cell" (GO:0097574).

- *Molecular Function ontology*: specifies the activities that the genes are involved in. Examples of terms from this ontology are: "protein transporter activity" (GO:0140318), "molecular carrier activity" (GO:0140104) and "binding" (GO:0005488).

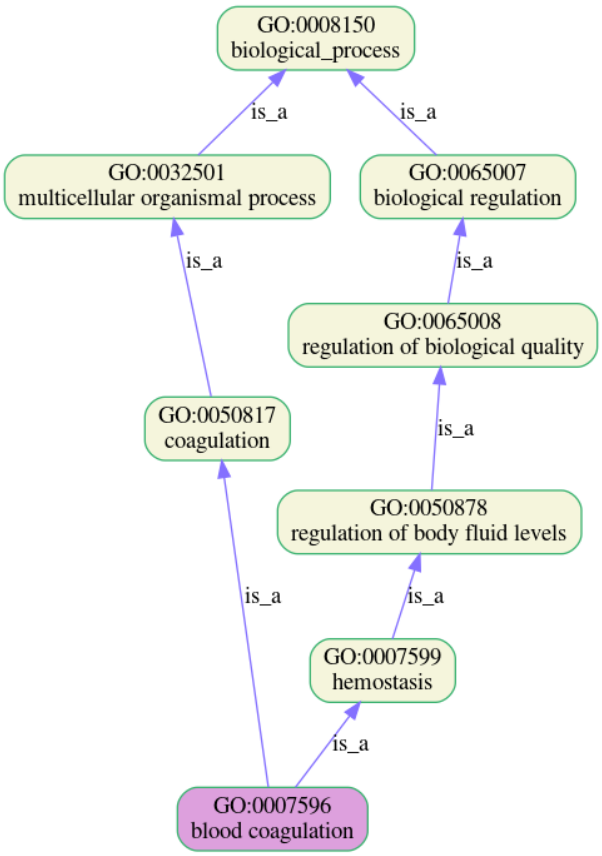
- *Biological Process ontology*: describes the biological processes that genes may take part in, which are defined by a sequence of molecular functions. Examples of terms from this ontology are: "reproduction" (GO:0000003), "immune response" (GO:0006955) and "response to drug" (GO:0042493).

After identifying the GO terms that are related to the genes of a bicluster, the significance (also called over-representation) of each term is measured by the Fisher's exact test (FISHER, 1922):

$$p = \sum_{i=t}^{b} \frac{\binom{c}{i}\binom{n-c}{b-i}}{\binom{n}{b}}, \tag{1.2}$$

where n is the total number of genes in the dataset (also called gene universe), c is the number of genes from the dataset that are annotated with the target category, b is the number of genes contained in the bicluster being analyzed, and t is the number of genes in the bicluster that are annotated with the target GO term. The p-value quantifies the probability of obtaining at least t genes annotated with a specific term in a random bicluster containing b genes (TANAY; SHARAN; SHAMIR, 2002). Usually, a term is considered over-represented if it achieves a p-value below 0.05 after the Benjamini and Hochberg multiple test correction (HOCHBERG; BENJAMINI, 1990).

Figure 3 – GO lineage of the term "blood coagulation" (GO:0007596).



Source – Image generated with the GOATOOLS package (TANG *et al.*, 2015).

1.2 The CRISPR system

In this section we introduce the main components of the CRISPR system, an adaptive immune system of bacterial and archaeal systems, as well as its immune mechanism stages. Such systems became widely known, after their potential as gene editing tools has been demonstrated.

1.2.1 Components and defense mechanism

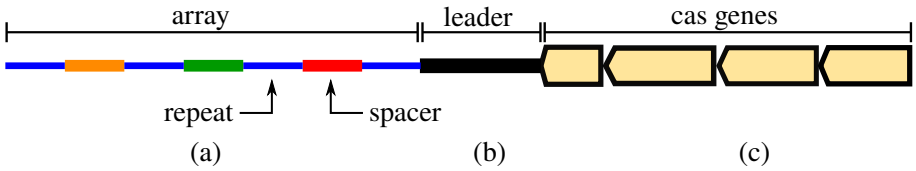
CRISPR systems are adaptive immune systems that are present in around 85% of archaeal and 40% of bacterial genomes, according to the recent study of Makarova *et al.* (2019). These systems are constituted by the following elements:

- *CRISPR array* (Figure 4a): which consists of short repeated nucleic acid sequences (called repeats) interleaved with other sequences of similar length (called spacers) that store information extracted from the nucleic acids of viruses that have either invaded the host organism or its ancestors (RATH *et al.*, 2015; ALKHNBASHI, 2017). When new viruses invade the organism, the spacers acquired from them are usually added next to the CRISPR

leader (Figure 4b). Thus, the CRISPR array can be seen as a chronological record of infections (RATH *et al.*, 2015; ALKHNBASHI, 2017).

- *CRISPR leader* (Figure 4b): which serves as the promoter region for the CRISPR array and is adjacent to it. The leader is composed by nucloetide sequences of low complexity that do not encode any protein (MOJICA; GARRETT, 2013; ALKHNBASHI, 2017).
- *CRISPR-associated (Cas) proteins* (Figure 4c): which are encoded by a set of adjacent *cas* genes and necessary for the CRISPR defense mechanism to work. This set of consecutive *cas* genes is usually called a cassette and the CRISPR system usually depends on different subsets of genes for each of the stages that perform the immunity function of this system (ALKHNBASHI, 2017). Such stages are described next.

Figure 4 – Illustration of the elements of the CRISPR system.



Source – Elaborated by the author.

In summary, the CRISPR system operates through three consecutive stages as discussed by Rath *et al.* (2015), Alkhnbashi (2017):

1. *Adaptation (or acquisition) stage*: where the information contained in the DNA of the invader is detected and stored as a new spacer in the CRISPR array. The acquisition of this new spacer can occur by: (i) naïve acquisition, when the invader is new and has not been encountered before, and (ii) priming acquisition, when the immune system has already encountered the invader before and there is one or more spacers that match it. In the latter case, the adaptation genes work in cooperation with the interference genes (which compose the last stage of the CRISPR system) to increase the resistance against recurring infections.
2. *Expression (or processing) stage*: where the information contained in the CRISPR array is transcribed into a long RNA molecule, called precursor CRISPR RNA (pre-crRNA), which is then processed into smaller molecules called CRISPR RNAs (crRNAs).
3. *Interference stage:* where the produced crRNAs associated with Cas proteins locate and cleave the foreign DNA element.

1.2.2 CRISPR classification

The *cas* genes that compose CRISPR systems belong to different families, present multiple variations even inside the same family and have a fast evolutionary behavior (MAKAROVA

et al., 2015; MAKAROVA *et al.*, 2019). To reflect such characteristics, many studies that extend the known CRISPR classifications have been published over the years. They aggregate the information that can be extracted from the increasing amount of data that becomes publicly available and the findings of diverse CRISPR studies that investigate different aspects of this system. Next, we summarize the main studies dedicated to these tasks.

Jansen *et al.* (2002) conducted a detailed investigation of the components of CRISPR systems that were found in 40 genomes of bacteria and archaea. In their analysis, the authors found 4 core *cas* genes, named cas1–cas4, which were frequently contained in the CRISPR cassettes found. Moreover, the authors observed some evidence that when multiple CRISPR cassettes are contained in the same genome, they may evolve independently.

Haft *et al.* (2005) identified a total of 45 different *cas* gene families across different archaeal and bacterial genomes. As a result, they defined 8 different CRISPR subtypes, based on the specific *cas* genes contained in each genome. The subtypes were named after the organisms where they were the only CRISPR instance detected: Ecoli, Ypest, Nmeni, Dvulg, Tneap, Hmari, Apern and Mtube. In addition, the authors also defined two new core *cas* genes, cas5 and cas6, which were associated with 5 and 4 of the CRISPR subtypes, respectively.

Makarova *et al.* (2011) improved previous studies by introducing a more complex categorization of CRISPR systems that took into account their evolutionary behavior. By analyzing 703 genomes, the authors proposed a hierarchical classification of CRISPR systems into 3 types (I, II and III), which could be decomposed into 10 subtypes (I-A to I-F, II-A and II-B, III-A and III-B), according to their gene combinations. Besides, they observed the presence of signature *cas* genes, which are responsible for defining the subtype of each cassette, and increased the size of the set of core *cas* genes by introducing cas7–cas10. Finally, the authors proposed the names I-U, II-U and III-U, for CRISPR systems that contained no signature genes, but could be assigned to one of the major types after performing structure and sequence analyses or even define new subypes in the future.

Makarova *et al.* (2015) analyzed 2,751 genomes and identified two new CRISPR (sub)types (IV and V). Additionally, the authors extended the previous CRISPR classification by establishing an extra top level in the classification hierarchy containing two classes (1 and 2). The main difference between them is that class 1 systems (containing types I, III and IV), which are the most common, rely on multiple Cas proteins for the interference stage, whereas class 2 systems (constituted by types II and V) depend on a single long Cas protein for it. Finally, the five types could be broken up into 16 subtypes (I-A to I-F, I-U, II-A to II-C, III-A to III-D, IV and V).

Shmakov *et al.* (2015) hypothesized that class 2 systems could be decomposed into more (sub)types, even though they are found in a much lesser extent when compared to class 1 systems. They took cas1 as an anchor to search for potentially new (sub)types, since it is one of the most common and conserved genes in CRISPR systems. As a result, 53 previously unclassified

CRISPR systems containing long proteins, which are one of the main characteristics of Type II systems, were identified. These systems could then be classified into two new subtypes (V-B and V-C)[3] and a new type (VI).

Makarova *et al.* (2019) collected 13,116 genomes and found 7,915 CRISPR systems. During their investigation, they identified 3 new class 1 and 13 new class 2 subtypes, increasing the CRISPR classification to 33 subtypes (I-A to I-G[4], II-A to II-C, III-A to III-F, IV-A to IV-C, V-A to V-I, V-K and VI-A to VI-D). Furthermore, supported by the new discoveries related to class 2 systems, the authors presented a hypothetical outline of the CRISPR evolution.

Finally, we show a simplified overview of the classes and types of the CRISPR system. Table 1 presents the combinations of Cas proteins that are involved in each stage of the CRISPR defense mechanism, separated by class and type. The discussion of the Cas protein composition and underlying differences of the mechanisms across the diverse CRISPR (sub)types is out of the scope of this thesis and we recommend the interested reader the recent paper of Makarova *et al.* (2019).

Table 1 – A simplified summary of the Cas proteins involved in each stage of the CRISPR system. Up to date, 13 core Cas proteins have been identified (Cas1–Cas13). The signal transduction/ancillary column refers to proteins whose roles are not certainly predicted.

Class	Type	Adaptation	Expression	Interference	Signal transduction/ ancillary
1	I	Cas1, Cas2, Cas4	Cas6	Cas3, Cas5, Cas7, Cas8, SS	–
	III	Cas1, Cas2, RT	Cas6	Cas5, Cas7, Cas10, SS	
	IV	Cas1, Cas2	Cas6	Cas5, Cas7, Csf1, SS†	DinG
2	II	Cas1, Cas2, Cas4	Rnase III*, Cas9	Cas9	Csn2
	V	Cas1, Cas2, Cas4	Cas12	Cas12	–
	VI	Cas1, Cas2	Cas13	Cas13	–

SS: small subunit. Usually, it is a Cas11 protein.
RT: reverse transcriptase. It is an enzyme responsible for the reverse transcription (Section 1.1.1).
† may be dispensable.
* non-Cas protein.

Source – Extracted from Box 1b of Makarova *et al.* (2019).

1.2.3 Applications

The CRISPR system has attracted a large deal of attention, mainly because of the Cas9 protein that is present in Type II systems. This protein can be used as a simple and effective approach for applications of genome engineering and gene editing (RAN *et al.*, 2013; SANDER; JOUNG, 2014; HSU; LANDER; ZHANG, 2014). Examples of applications that may benefit from the use of CRISPR/Cas9 are: development of new cancer (ZHAN *et al.*, 2019) and HIV (EBINA *et al.*, 2013) therapies; correction of genetic diseases (WU *et al.*, 2013); design of

[3] Shmakov *et al.* (2015) redesignated the type V systems identified by Makarova *et al.* (2015) as subtype V-A.

[4] Subtype I-U was reclassified as I-G.

better crops with increased resistance against climate changes (KHATODIA *et al.*, 2016); among others.

1.3 Main contributions

After providing a general overview of the areas of the Bioinformatics covered by this thesis, we present the main contributions of our work.

This thesis can be divided in two parts. The first one is directly related to the original research project that was submitted to the ICMC/USP as the application for the PhD course. Initially, we were planning on improving the evaluation of biclustering algorithms on real gene expression datasets, by investigating the behavior of bicluster coherence measures. However, in 2018 we had the opportunity to collaborate with the Bioinformatics Group of the University of Freiburg. In this collaboration, we developed ML-based tools to solve CRISPR-related challenges. The results of this collaboration constitute the second part of this thesis. In summary, our contributions are:

- Regarding bicluster coherence measures (first part):
  - A broad analysis of 17 bicluster coherence measures and their correlations with GO over-representations.
  - Time complexity analyses of the bicluster coherence measures, which we could not find in the literature.
  - The evaluation of 16 different experimental scenarios involving bicluster coherence measures, 2 correlation coefficients and all three gene ontologies.
- Regarding CRISPR-related challenges (second part):
  - A novel tool for the classification of CRISPR-Cas systems.
  - The prediction of possibly missing Cas proteins inside CRISPR cassettes.
  - A set of association rules that indicate Cas proteins that are biologically related.
  - A new approach for identifying cassette boundaries inside a genome and labeling the respective *cas* genes.
  - A study case for the detection of putative new *cas* gene types.

1.4 Thesis organization

This thesis is organized as a collection of papers. The first part (chapter 2) and the second part (chapters 3 and 4) reproduce the entire content of three papers that were published

in international journals. They can be read in any particular order, as preferred by the reader, since each one is self-contained with all the background necessary to understand it. The only exception is chapter 5, which concludes this thesis. Below we briefly describe each chapter as well as its main contributions.

1.4.1 Chapter 2

Title: "*Experimental correlation analysis of bicluster coherence measures and gene ontology information*". A preliminary version of this chapter was published in the Proceedings of the 7th Brazilian Conference on Intelligent Systems (BRACIS 2018) (PADILHA; CARVALHO, 2018), for which we received an honorable mention and were invited to extend it for publication in the Applied Soft Computing journal by Elsevier. This chapter is this extended article (PADILHA; CARVALHO, 2019).

We present a comparative study of 17 bicluster coherence measures from the literature and analyze their behavior with respect to external knowledge extracted from the gene ontologies. They are applied on the results obtained by 10 biclustering algorithms on a benchmark of 19 gene expression datasets. In summary, the contributions of this chapter are:

- The analysis of the correlation between the measures on real gene expression data, allowing the identification of redundant measures.
- An extensive evaluation of scenarios involving the Biological Process, Cellular Component and Molecular Function gene ontologies.
- The time complexity analyses of the evaluated measures.

1.4.2 Chapter 3

Title: "*CRISPRcasIdentifier: Machine learning for accurate identification and classification of CRISPR-Cas systems*". This chapter is an article that was written in collaboration with Dr. Omer S. Alkhnbashi and Prof. Dr. Rolf Backofen from the Bioinformatics Group of the University of Freiburg, Germany, and Dr. Shiraz A. Shah from COPSAC, University of Copenhagen, Denmark. It was published in the GigaScience journal by Oxford University Press (PADILHA *et al.*, 2020a).

We introduce a holistic approach named CRISPRcasIdentifier, which uses ML models to predict missing Cas proteins in CRISPR cassettes and their subtypes according to the most recent CRISPR data publicly available (MAKAROVA *et al.*, 2015; MAKAROVA *et al.*, 2019). In addition, our tool is also able to extract association rules constituted by Cas proteins that frequently co-occur in CRISPR systems and may constitute a functional module. In summary, the contributions of this chapter are:

- The first approach for predicting possibly missing proteins in CRISPR cassettes.
- The classification of CRISPR subtypes with a high accuracy, clearly outperforming the competing tools from the literature.
- The extraction of association rules from CRISPR-Cas systems, which indicate sets of Cas proteins that frequently co-occur and are thus biologically related.
- The identification of signature genes from CRISPR systems based on the rules of decision trees and on the importance of the Cas proteins for ensembles of decision trees.

1.4.3 Chapter 4

Title: "*Casboundary: Automated definition of integral Cas cassettes*". This chapter is an article that was written in collaboration with Dr. Omer S. Alkhnbashi, Dr. Van Dinh Tran and Prof. Dr. Rolf Backofen from the Bioinformatics Group of the University of Freiburg, Germany, and Dr. Shiraz A. Shah from COPSAC, University of Copenhagen, Denmark. It was published in the Bioinformatics journal by Oxford University Press (PADILHA *et al.*, 2020c).

We propose a tool for automatic detection of cassette boundaries in archaeal and bacterial genomes. Besides, it is also able to classify the *cas* genes of the cassette and to decompose it into the different modules that constitute the CRISPR system. In summary, the contributions of this chapter are:

- The mathematical formulation of the cassette boundary identification problem.
- The first tool that is able to detect cassette boundaries by taking the relations between a potential signature gene and its neighbors into account.
- An approach for the classification of *cas* gene types that combines multiple features based on Hidden Markov Models and properties of the respective proteins.
- The detection of potentially new *cas* gene types and a study case that shows the identification of putative new gene families.
- The decomposition of a cassette into its modules (adaptation, expression and interference).
- A tool that, when integrated with CRISPRcasIdentifier, provides a full pipeline for the detection of CRISPR cassettes, as well as *cas* type and cassette subtype classification.

1.4.4 Chapter 5

In this chapter we present our main conclusions, discuss some limitations and present the directions of some possible future studies.

CHAPTER

2

EXPERIMENTAL CORRELATION ANALYSIS OF BICLUSTER COHERENCE MEASURES AND GENE ONTOLOGY INFORMATION

Publication information: This chapter is an article that was published in the Applied Soft Computing journal by Elsevier. According to the publisher's policies, we retain the right to include it in this thesis for non-commercial purposes*.

Reference: PADILHA, V. A.; CARVALHO, A. C. P. L. F. Experimental correlation analysis of bicluster coherence measures and gene ontology information. **Applied Soft Computing**, Elsevier, v. 85, p. 105688, 2019. Available: <https://doi.org/10.1016/j.asoc.2019.105688>.

2.1 Abstract

Biclustering algorithms have become popular tools for gene expression data analysis. They can identify local patterns defined by subsets of genes and subsets of samples, which cannot be detected by traditional clustering algorithms. In spite of being useful, biclustering is an NP-hard problem. Therefore, the majority of biclustering algorithms look for biclusters optimizing a pre-established coherence measure. Many heuristics and validation measures have been proposed for biclustering over the last 20 years. However, there is a lack of an extensive comparison of bicluster coherence measures on practical scenarios. To deal with this lack, this paper experimentally analyzes 17 bicluster coherence measures and external measures calculated from information obtained in the gene ontologies. In this analysis, results were produced by 10 algorithms from the literature in 19 gene expression datasets. According to the experimental results, a few pairs of strongly correlated coherence measures could be identified, which suggests redundancy. Moreover, the pairs of strongly correlated measures might change when dealing

*<https://www.elsevier.com/about/policies/copyright>

with normalized or non-normalized data and biclusters enriched by different ontologies. Finally, there was no clear relation between coherence measures and assessment using information from gene ontology.

2.2 Introduction

High-throughput technologies, such as microarrays (BROWN; BOTSTEIN, 1999), allow researchers to monitor the behavior of thousands of genes under specific biological samples. Normally, the samples correspond to different points in a time series, different types of tissues, different environmental conditions, different organs and/or different individuals (MADEIRA; OLIVEIRA, 2004).

Gene expression data analysis studies investigate the behavior of thousands of genes from an organism under multiple biological samples. Their results and conclusions can support a better understanding of gene functions, biological processes, effects of treatments, among others (BEN-DOR; SHAMIR; YAKHINI, 1999; TANAY; SHARAN; SHAMIR, 2002). For such, these studies use a data matrix representation, which is obtained by concatenating the results from multiple high-throughput experiments. In such a matrix, each row corresponds to a gene, each column corresponds to a sample and each element quantifies the expression level of a gene in a specific sample (MADEIRA; OLIVEIRA, 2004; JIANG; TANG; ZHANG, 2004).

Traditional clustering algorithms are often used to analyze gene expression data. They allow researchers to improve their understanding of the functions of the genes from an organism. However, some studies argue that a biological process may be active only under subsets of genes and subsets of samples (CHENG; CHURCH, 2000; MADEIRA; OLIVEIRA, 2004; PONTES; GIRLDEZ; AGUILAR-RUIZ, 2015), which characterizes clusters in subspaces of the original dataset. Besides, some genes or samples may not take part in any cluster at all. Thus, a traditional clustering method may not be able to answer some important research questions.

Biclustering overcomes the previously discussed clustering limitations. It looks for local patterns, called biclusters, comprising subsets of genes and subsets of samples, which are usually obfuscated by the high dimensionality of a dataset. Additionaly, biclustering may allow the presence of overlapped biclusters and unclustered genes or samples.

Although biclustering has proved its importance, the size of its search space is in the order of 2^{N+M} for N genes and M samples, which characterizes an NP-hard problem (CHENG; CHURCH, 2000; TANAY; SHARAN; SHAMIR, 2002; MADEIRA; OLIVEIRA, 2004; JIANG; TANG; ZHANG, 2004). Therefore, many algorithms are based on the optimization of a bicluster coherence measure through (meta-)heuristics, in order to produce approximate results in an acceptable amount of time.

The selection or proposal of an appropriate coherence measure is crucial in the develop-

ment of a biclustering algorithm (PONTES; GIRLDEZ; AGUILAR-RUIZ, 2015). Each measure can detect a particular set of patterns and it is the main component that guides the algorithm search for a good solution.

Since 2000, many biclustering algorithms and coherence measures have been proposed in the literature. At the same time, extensive surveys (MADEIRA; OLIVEIRA, 2004; TANAY; SHARAN; SHAMIR, 2005; BUSYGIN; PROKOPYEV; PARDALOS, 2008; PONTES; GIRÁLDEZ; AGUILAR-RUIZ, 2015) and studies for the comparison of algorithms were carried out (PRELIĆ *et al.*, 2006; BOZDAĞ; KUMAR; CATALYUREK, 2010; EREN *et al.*, 2012; OGHABIAN *et al.*, 2014; PADILHA; CAMPELLO, 2017). However, few studies have investigated the behaviors of different coherence measures and to what extent they agree with the external biological information available.

In a preliminary study (PADILHA; CARVALHO, 2018), we investigated the correlations of 15 biclustering coherence measures for results generated by 9 biclustering algorithms in 19 gene expression datasets. For such, we considered two experimental scenarios on normalized data to analyze relations between coherence criteria and biological significance of biclusters. The present study extends this work, presenting the following contributions:

- Analysis of correlations between 17 coherence measures for results obtained by 10 biclustering algorithms in the 19 gene expression datasets, to present evidence able to reduce the use of redundant measures during evaluation;

- Evaluation of results for 16 different experimental scenarios, which encompass normalized and non-normalized data, separate and aggregated analyses with the available ontologies. Then, we have more evidence to assess if the performances of coherence measures agree with those achieved by evaluation using external knowledge; and

- Computational complexity analyses of the measures, which are usually not provided in their original studies, and were only provided in the supplementary material of (PADILHA; CARVALHO, 2018). These analyses are important for applications where a large number of biclusters needs to be assessed.

This paper is organized as follows. Section 2.3 presents the main related works found by the authors. Section 2.4 introduces the coherence and external measures, biclustering algorithms and the gene expression datasets selected for this study. Section 2.5 presents the experiments carried out and discusses their results. Finally, Section 2.6 presents the main conclusions from this study.

2.3 Related work

There are several studies which propose new biclustering algorithms and/or coherence measures. However, there is a lack of extensive comparisons between distinct measures on the results of different algorithms. For instance, few related studies discuss how biclustering coherence measures relate to each other.

The first study of biclustering in the context of gene expression data can be found in Cheng and Church (2000). The well-known Mean Squared Residue (MSR) coherence measure and an algorithm for its optimization were proposed. According to the experimental results obtained, the biclustering paradigm used affects the gene expression data analysis. This study became the main benchmark adopted when developing new bicluster coherence measures and algorithms.

In Aguilar-Ruiz (2005), the authors formally analyze the MSR measure, showing its limitation for identifying scaling patterns in biclusters, due to its high dependence on the variances of scaling factors. Other studies, such as Divina *et al.* (2012), Giraldez *et al.* (2007), Teng and Chan (2008), Mukhopadhyay, Maulik and Bandyopadhyay (2009), Pontes, Giráldez and Aguilar-Ruiz (2010), Nepomuceno, Troncoso and Aguilar-Ruiz (2011), Flores *et al.* (2013), introduced bicluster coherence measures able to overcome the limitations of MSR on certain types of patterns. The improvements obtained were shown in the performance on synthetic and/or real data.

In Santamaría, Quintales and Therón (2007), the authors proposed a coherence measure and an internal biclustering evaluation index. They also discussed the main advantages and disadvantages when using relative, internal and external measures. The authors tested their proposals on synthetic datasets for the task of hyperparameter selection for two biclustering algorithms.

A new measure, the Minimal Mean Squared Error (MMSE), to detect linear patterns in biclusters, was proposed in Chen, Liu and Zeng (2015). The authors compared MMSE with five measures from the literature. They also adapted the algorithm proposed in Cheng and Church (2000) to optimize the new measure and performed experiments on synthetic and real datasets. This modified algorithm was compared with 6 biclustering and 2 clustering algorithms from the literature. According to the authors, the new measure and algorithm detected patterns not usually found by other measures and algorithms.

A large number of coherence measures, 14 altogether, were discussed in Pontes, Girldez and Aguilar-Ruiz (2015). From these 14 measures, 13 were tested on synthetic datasets and on 4 real datasets. In the experiments with synthetic datasets, to assess these coherence measures when the biclusters do not follow perfect patterns, they were tested on 3 types of bicluster patterns subject to different noise levels. In the experiments with real datasets, the measures were applied to biclusters found by an evolutionary algorithm previously proposed. However, as this

algorithm includes one of the investigated measures in its fitness function, there is a bias in the experimental results. Afterwards, the coherence measures were compared to values obtained from external biological information. According to the authors, the correlation between their results and the biological measures according to a normalized Mutual Information (MI) score showed a relation between many coherence measures with the biological information.

This paper goes one step forward in the previous analysis by comparing a larger number of 17 biclustering measures in a larger number of datasets (19). Besides, in order to reduce algorithmic bias, each measure was evaluated using 10 biclustering algorithms. In order to reduce dependence on estimation algorithms or on data binning to calculate MI for continuous variables, the Pearson and Spearman correlations were used. Additionally, the Wilcoxon signed-rank test was applied to the results to assess any evidence of differences between the results from the two correlation measures.

2.4 Methods

This section has a description of the main methods used. For such, it is organized as follows. In Section 2.4.1, we discuss the types of numeric bicluster patterns. In Section 2.4.2, we present the coherence measures investigated. In Section 2.4.3, we describe the 10 algorithms used in the experiments. In Section 2.4.4, we present the 19 datasets selected. In Section 2.4.5, we discuss the external evaluation using GO ontologies and the quantities calculated from them. In Section 2.4.6, we detail our experimental methodology. Finally, in Section 2.4.7, we describe the hyperparameter settings used for the algorithms.

2.4.1 Bicluster patterns

Let $X = (R, C)$ be a gene expression matrix, where R is a set of N rows (genes) and C is a set of M columns (samples). A bicluster corresponds to a submatrix $B = (I, J)$, $I \subseteq R$, $J \subseteq C$, which presents some patterns between its values. Several numeric patterns have been described in the literature. The most general among them are (PONTES; GIRÁLDEZ; AGUILAR-RUIZ, 2015):

- *Shifting pattern*, where each bicluster element b_{ij} can be defined by a constant/typical value π_i for the i^{th} row added to an adjustment factor β_j for the j^{th} column. Thus, $b_{ij} = \pi_i + \beta_j$.
- *Scaling pattern*, where each bicluster element b_{ij} is described by the constant/typical value π_i for the i^{th} row multiplied by an adjustment factor α_j for the j^{th} column. Thus, $b_{ij} = \pi_i \alpha_j$.
- *Shifting-scaling pattern*, where the bicluster presents both patterns simultaneously. Each bicluster element b_{ij} is obtained by multiplying π_i by α_j and adding the result to β_j. Thus,

$b_{ij} = \pi_i \alpha_j + \beta_j$. Note that shifting and scaling biclusters are special cases of shifting-scaling patterns when $\alpha_j = 1$ and $\beta_j = 0 \; \forall j \in J$, respectively.

From the aforementioned patterns, some specific patterns that are also widely referenced in the literature can be extracted, such as (MADEIRA; OLIVEIRA, 2004):

- *Constant pattern*, where all of the bicluster elements are equal to the same constant value μ. Thus, $\pi_i = \mu \; \forall i \in I$, $\alpha_j = 1$ and $\beta_j = 0 \; \forall j \in J$.
- *Constant row pattern*, where the elements of each row of the bicluster are equal to the same constant value, which can be different from one row to another. Thus, $\alpha_j = 1$ and $\beta_j = 0 \; \forall j \in J$.
- *Constant column pattern*, where the elements of each column of the bicluster are equal to the same constant value, which can be different from one column to another. Thus, $\pi_i = 1$ $\forall i \in I$.

It must be mentioned that, in real gene expression data, the expression values may be obfuscated by the presence of noise. Therefore, one cannot expect the biclusters to always present the perfect patterns previously described. Thus, for each element x_{ij} of the original data matrix X, there is generally an unknown η_{ij} value associated to it, which represents its amount of noise (MADEIRA; OLIVEIRA, 2004). This motivates the use of coherence measures, which quantify the extent of agreement between a noisy bicluster and a desired ideal pattern.

2.4.2 Coherence measures

The use of coherence measures is an important step to evaluate a set of biclusters that were produced by one or more biclustering algorithms. These measures require only the data available and inspect the quality of the biclusters' elements regarding a set of predefined patterns. By using different measures, the results can be assessed from different perspectives and, as a consequence, cover different aspects of the data based on distinct approaches, such as: the variability of the bicluster's values (Variance-based), correlations among genes or biological samples (Correlation-based), and correspondence of the bicluster's elements with a general tendency pattern that models their behavior (Standardization-based).

In this section, we introduce the coherence measures investigated in this paper. They are the same measures investigated in Pontes, Girldez and Aguilar-Ruiz (2015) and four additional measures which, to the best of our knowledge, have not been previously investigated in related studies: three were the main contributions of the bicluster evaluation work in Santamaría, Quintales and Therón (2007) that assesses constant patterns, constant row patterns and constant column patterns; the fourth, proposed in Chen, Liu and Zeng (2015), can capture shifting-scaling

biclusters that, although is the most general bicluster model discussed in the literature, is the hardest one to deal with and only few measures are able to properly evaluate it.

Next, we present the measures, organized in the following categories, according to the similarities of their approaches: Variance-based (Section 2.4.2.1), Correlation-based (Section 2.4.2.2) and Standardization-based (Section 2.4.2.3). We also provide the time complexity analyses for the measures, which are usually not provided in their original publications. In Table 2, we present a summary of the measures: range of values, objectives (i.e., if a measure must be maximized or minimized) and time complexity.

2.4.2.1 Variance-based measures

The measures from this category evaluate the coherence of the values of a bicluster regarding their expected values predicted using quantities, such as the bicluster mean or the bicluster row and column means. In this paper, b_{iJ}, b_{Ij} and b_{IJ} stand for the mean of the i^{th} row, the mean of the j^{th} column and the mean of all elements of a bicluster B, respectively. These measures are presented next.

1. *Variance (VAR)* (HARTIGAN, 1972) is used to detect constant patterns:

$$\mathrm{VAR}(B) = \sum_{i=1}^{|I|}\sum_{j=1}^{|J|}(b_{ij} - b_{IJ})^2. \tag{2.1}$$

   Clearly, the smaller the value, the closer a bicluster is to a constant pattern.

   Time complexity analysis. The calculation of b_{IJ} costs $O(|I||J|)$. The sum of the squared terms also costs $O(|I||J|)$. Overall, the time complexity of VAR is $O(|I||J|)$.

2. *Mean Squared Residue (MSR)* (CHENG; CHURCH, 2000) is based on the shifting bicluster model, and produces smaller values for biclusters that agree more with this model. MSR is defined as:

$$\mathrm{MSR}(B) = \frac{1}{|I||J|}\sum_{i=1}^{|I|}\sum_{j=1}^{|J|}(b_{ij} - b_{iJ} - b_{Ij} + b_{IJ})^2. \tag{2.2}$$

   Time complexity analysis. The calculations of $b_{iJ}\,\forall i \in I$, $b_{Ij}\,\forall j \in J$ and b_{IJ} require $O(|I||J|)$ steps. The sum of all squared terms also requires $O(|I||J|)$ steps. Overall, the time complexity of MSR is $O(|I||J|)$.

3. *Mean Absolute Residue (MAR)* (YANG *et al.*, 2002) is also based on the shifting bicluster model. The only difference between MAR and MSR is that MAR takes the absolute difference between the bicluster elements and their expected values predicted by the row, column and bicluster means. It is defined as:

$$\mathrm{MAR}(B) = \frac{1}{|I||J|}\sum_{i=1}^{|I|}\sum_{j=1}^{|J|}|b_{ij} - b_{iJ} - b_{Ij} + b_{IJ}|. \tag{2.3}$$

Time complexity analysis. MAR has the same time of complexity of MSR, which is $O(|I||J|)$.

4. *Relevance Index (RI)* (YIP; CHEUNG; NG, 2004) identifies the constant columns pattern based on the local and global variances of the columns in the bicluster. It is formulated as:

$$\mathrm{RI}(B) = \sum_{j=1}^{|J|} R_j, \tag{2.4}$$

where

$$R_j = 1 - \frac{\sigma_{Ij}^2}{\sigma_j^2}, \tag{2.5}$$

σ_{Ij}^2 is the variance of the j^{th} column of B and σ_j^2 is the variance of the j^{th} column of the full dataset.

Time complexity analysis. The calculation of each σ_{Ij}^2 costs $O(|I|)$. The calculation of each σ_j^2 costs $O(N)$. Therefore, any R_j requires $O(|I|) + O(N) = O(N)$ steps. Since B has $|J|$ columns, the complexity of RI is $O(N|J|)$.

5. *Constancy by rows (C_r)* (SANTAMARÍA; QUINTALES; THERÓN, 2007) quantifies the agreement of a bicluster with the constant column pattern:

$$C_r(B) = \frac{1}{|I|} \sum_{i=1}^{|I|-1} \sum_{k=i+1}^{|I|} \sqrt{\sum_{j=1}^{|J|} (b_{ij} - b_{kj})^2}. \tag{2.6}$$

Time complexity analysis. The sum of the squared terms costs $O(|J|)$. In a bicluster, there is a total of $(|I|(|I|-1))/2 = O(|I|^2)$ pairs of rows. Overall, C_r runs in $O(|I|^2|J|)$.

6. *Constancy by columns (C_c)* (SANTAMARÍA; QUINTALES; THERÓN, 2007) expresses the extent to which the values of a bicluster present a constant row pattern. It is the transposed version of C_r.

Time complexity analysis. Since C_c is the transposed version of C_r, its time complexity is $O(|I||J|^2)$.

7. *Overall Constancy (OC)* (SANTAMARÍA; QUINTALES; THERÓN, 2007) minimizes its value when evaluating constant biclusters. For such, it integrates the constancy by rows and the constancy by columns formulae:

$$\mathrm{OC}(B) = \frac{|I|C_r(B) + |J|C_c(B)}{|I| + |J|}. \tag{2.7}$$

Time complexity analysis. Since it requires the calculation of C_r and C_c, OC runs in $O(\max(|I|^2|J|, |I||J|^2))$.

8. *Scaling Mean Squared Residue (SMSR)* (MUKHOPADHYAY; MAULIK; BANDYOPADHYAY, 2009) is a modification of the MSR measure that is able to detect scaling biclusters:

$$\text{SMSR}(B) = \frac{1}{|I||J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} \frac{(b_{iJ} b_{Ij} - b_{ij} b_{IJ})^2}{b_{iJ}^2 b_{Ij}^2}. \tag{2.8}$$

As in MSR, smaller values indicate biclusters that better suit the desired model.

Time complexity analysis. SMSR requires the same quantities as MSR and MAR ($b_{iJ} \forall i \in I$, $b_{Ij} \forall j \in J$ and b_{IJ}) to determine the differences among the values of the bicluster elements and their expected values. Therefore, the complexity of SMSR is $O(|I||J|)$.

9. *Minimal Mean Squared Error (MMSE)* (CHEN; LIU; ZENG, 2015) is based on the shifting, scaling and shifting-scaling models. Its authors argue that it is better suited than previous measures, such as MSR and SMSR, to identify negative correlated linear patterns. This measure is formally expressed as:

$$\text{MMSE}(B) = \frac{1}{|I||J|} \left[\sum_{i=1}^{|I|} \sum_{j=1}^{|J|} d_{ij}^2 - \lambda_{\max}(DD^T) \right], \tag{2.9}$$

where $d_{ij} = b_{ij} - b_{iJ}$, D is the matrix containing all d_{ij} elements, and $\lambda_{\max}(DD^T)$ is the eigenvalue of DD^T with maximum absolute value.

The time complexity of MMSE is $O(\min(|I|, |J|)\ |I||J|)$. The complete analysis is provided in the original paper.

2.4.2.2 Correlation-based measures

These measures assess the similarity between gene/sample behaviors, instead of the magnitudes or deviations among their values (PONTES; GIRLDEZ; AGUILAR-RUIZ, 2015). For such, they use either the Pearson or the Spearman correlation to measure gene/sample similarities. In this paper, the former is denoted as $r(\cdot,\cdot)$ while the latter is represented as $\rho(\cdot,\cdot)$. In addition, the i^{th} row and the j^{th} column of B are denoted as b_{i*} and b_{*j}, respectively. These measures are detailed next.

1. *Average Correlation (AC)* (NEPOMUCENO; TRONCOSO; AGUILAR-RUIZ, 2011) was proposed to detect shifting, scaling and shifting-scaling biclusters by calculating the average Pearson correlation between its rows:

$$\text{AC}(B) = \frac{2}{|I|(|I|-1)} \sum_{i=1}^{|I|-1} \sum_{k=i+1}^{|I|} r(b_{i*}, b_{k*}). \tag{2.10}$$

Time complexity analysis. The calculation of each $r(b_{i*}, b_{k*})$ costs $O(|J|)$. There are $|I|\,(|I|-1)\,/\,2 = O(|I|^2)$ pairs of rows in B. Overall, AC requires $O(|I|^2\,|J|)$ steps.

2. *Sub-matrix Correlation Score (SCS)* (YANG; DAI; YAN, 2011) was proposed to detect shifting or scaling patterns. It takes into account correlations between rows and between columns. The ideal bicluster would present strong correlations on both dimensions. SCS is formally defined as:

$$\mathrm{SCS}(B) = \min\{S_{\mathrm{row}}(B), S_{\mathrm{col}}(B)\}, \tag{2.11}$$

where

$$S_{\mathrm{row}}(B) = \min_{i=1,\cdots,|I|} \left\{ 1 - \frac{1}{|I|-1} \sum_{\substack{k=1 \\ k\neq i}}^{|I|} |r(b_{i*}, b_{k*})| \right\}, \tag{2.12}$$

$$S_{\mathrm{col}}(B) = \min_{j=1,\cdots,|J|} \left\{ 1 - \frac{1}{|J|-1} \sum_{\substack{l=1 \\ l\neq j}}^{|J|} |r(b_{*j}, b_{*l})| \right\}. \tag{2.13}$$

Time complexity analysis. The calculation of each $r(b_{i*}, b_{k*})$ and each $r(b_{*j}, b_{*l})$ costs $O(|J|)$ and $O(|I|)$, respectively. The calculations of all S_{row} values and all S_{col} values require $O(|I|^2\,|J|)$ and $O(|I|\,|J|^2)$ steps, respectively. Overall, the time complexity of SCS is $O(\max(|I|^2\,|J|, |I|\,|J|^2))$.

3. *Average Correlation Value (ACV)* (TENG; CHAN, 2008) was designed to identify shifting or scaling models. For such, it gives higher values for biclusters containing rows or columns presenting a strong average Pearson correlation value:

$$\mathrm{ACV}(B) = \max \left\{ \begin{array}{l} \dfrac{2}{|I|(|I|-1)} \displaystyle\sum_{i=1}^{|I|-1} \sum_{k=i+1}^{|I|} |r(b_{i*}, b_{k*})|, \\ \dfrac{2}{|J|(|J|-1)} \displaystyle\sum_{j=1}^{|J|-1} \sum_{l=j+1}^{|J|} |r(b_{*j}, b_{*l})| \end{array} \right\}. \tag{2.14}$$

Time complexity analysis. The average absolute correlation among the rows of B requires $O(|I|^2\,|J|)$ steps. The average absolute correlation between the columns of B costs $O(|I|\,|J|^2)$. Therefore, ACV runs in $O(\max(|I|^2\,|J|, |I|\,|J|^2))$.

4. *Average Spearman's Rho (ASR)* (AYADI; ELLOUMI; HAO, 2009) was proposed to overcome any sensitivity of the ACV measure due to using the Pearson correlation. It is formulated as:

$$\mathrm{ASR}(B) = \max \left\{ \begin{array}{l} \dfrac{2}{|I|(|I|-1)} \displaystyle\sum_{i=1}^{|I|-1} \sum_{k=i+1}^{|I|} \rho(b_{i*}, b_{k*}), \\ \dfrac{2}{|J|(|J|-1)} \displaystyle\sum_{j=1}^{|J|-1} \sum_{l=j+1}^{|J|} \rho(b_{*j}, b_{*l}) \end{array} \right\}. \tag{2.15}$$

Time complexity analysis. The Spearman coefficient measures the correlation between the ranks of the elements of two vectors. For such, it requires a sorting step, which can be

performed in $O(n \log n)$ for n elements. Thus, each $\rho(b_{i*}, b_{k*})$ and each $\rho(b_{*j}, b_{*l})$ cost $O(|J| \log |J|)$ and $O(|I| \log |I|)$, respectively. The first argument of max runs in $O(|I|^2 |J| \log |J|)$. The latter argument of max requires $O(|J|^2 |I| \log |I|)$ steps. Overall, ASR runs in $O(\max(|I|^2 |J| \log |J|, |J|^2 |I| \log |I|))$.

5. *Spearman's Biclustering Measure (SBM)* (FLORES *et al.*, 2013) was introduced to detect shifting or scaling patterns by calculating the average Spearman correlation coefficient between the rows and columns of a bicluster and weighting their influences in the final result. Formally, this measure is defined as:

$$\text{SBM}(B) = \psi(B)\ \omega(B)\ \bar{\rho}_I(B)\ \bar{\rho}_J(B), \tag{2.16}$$

where

$$\bar{\rho}_I(B) = \frac{2}{|I|(|I|-1)} \sum_{i=1}^{|I|-1} \sum_{k=i+1}^{|I|} |\rho(b_{i*}, b_{k*})|, \tag{2.17}$$

$$\bar{\rho}_J(B) = \frac{2}{|J|(|J|-1)} \sum_{j=1}^{|J|-1} \sum_{l=j+1}^{|J|} |\rho(b_{*j}, b_{*l})|, \tag{2.18}$$

$\psi(B)$ and $\omega(B)$ are hyperparameters that refer to the importance of the rows and the columns of a bicluster. Their values are set by the user. In this paper, we used $\omega(B) = 1$ and

$$\psi(B) = \begin{cases} 1, & \text{if } |J| > 9, \\ \dfrac{|J|}{M}, & \text{otherwise,} \end{cases} \tag{2.19}$$

which are the default values used by the original authors.

Time complexity analysis. SBM is calculated in constant time after $\bar{\rho}_I(B)$ and $\bar{\rho}_J(B)$ are obtained. Therefore, SBM has the same time complexity of ASR: $O(\max(|I|^2 |J| \log |J|, |J|^2 |I| \log |I|))$.

In Pontes, Girldez and Aguilar-Ruiz (2015), the Pearson correlation was also included in this category. However, this measure can only be calculated between pairs of rows or pairs of columns and not for a whole bicluster. The authors did not mention how they summarized all the Pearson correlation values for gene or sample pairs of a bicluster. The most simple approach would be to return the average value. However, this is exactly what the AC measure does. For this reason, the Pearson correlation was not considered as a bicluster coherence measure by itself in this study.

2.4.2.3 Standardization-based measures

These coherence measures are based on standardization evaluation of the bicluster's rows/columns tendencies by scaling their values to make them comparable (PONTES; GIRLDEZ;

AGUILAR-RUIZ, 2015). Thus, these measures are calculated on the standardized bicluster B', whose elements are defined as:

$$b'_{ij} = \frac{b_{ij} - \mu_i}{\sigma_i}, \tag{2.20}$$

where μ_i and σ_i are the mean and the standard deviation of the i^{th} row (gene) of B, respectively. These measures are detailed next.

1. *Maximal Standard Area (MSA)* (GIRALDEZ *et al.*, 2007) defines a band for the set of columns of a bicluster, which corresponds to the maximum and minimum values of each column. The value of MSA is the total area of this band. This measure, which has been applied to detect shifting or scaling bicluster patterns, is defined as:

$$\text{MSA}(B) = \sum_{j=1}^{|J|-1} \left| \frac{\max_j^{B'} - \min_j^{B'} + \max_{j+1}^{B'} - \min_{j+1}^{B'}}{2} \right|, \tag{2.21}$$

   where $\max_j^{B'}$ and $\min_j^{B'}$ correspond to the maximum and minimum values of the jth column of B', respectively.

   Time complexity analysis. $\max_j^{B'}$ and $\min_j^{B'}$ require $O(|I|)$ steps. Since we have $|J|$ columns in the bicluster, MSA runs in $O(|I|\,|J|)$.

2. *Virtual Error (VE)* (DIVINA *et al.*, 2012) calculates the difference between the bicluster elements and a virtual row (gene) pattern that captures the general trend of the bicluster values (PONTES; GIRLDEZ; AGUILAR-RUIZ, 2015). It is minimized when evaluating biclusters with shifting or scaling patterns. This measure is defined as:

$$\text{VE}(B) = \frac{1}{|I||J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} |b'_{ij} - p'_j|, \tag{2.22}$$

   where p is the mean row vector of B, and p' is its standardized version.

   Time complexity analysis. p requires $O(|I||J|)$ steps to be calculated. The standardization of B takes $O(|I||J|)$ steps. The standardization of p costs $O(|J|)$. The absolute differences between the elements of B' and the elements of p' require $O(|I||J|)$. Overall, VE runs in $O(|I||J|)$.

3. *Transposed Virtual Error (VEt)* (PONTES; GIRÁLDEZ; AGUILAR-RUIZ, 2010) is the VE measure applied in B^T. It is able to detect all the patterns identified by VE and also the shifting-scaling pattern.

   Time complexity analysis. VEt requires the same number of steps as VE: $O(|I||J|)$.

2.4.3 Algorithms

To investigate the behavior of the coherence measures, we selected 10 biclustering algorithms often used in the literature, which have already been extensively studied and have

Table 2 – Summary of the investigated measures.

Category	Measure	Reference	Range	Objective	Time complexity
Variance-based	VAR	(HARTIGAN, 1972)	$[0,\infty)$	Min	$O(\lvert I\rvert \lvert J\rvert)$
	MSR	(CHENG; CHURCH, 2000)	$[0,\infty)$	Min	$O(\lvert I\rvert \lvert J\rvert)$
	MAR	(YANG *et al.*, 2002)	$[0,\infty)$	Min	$O(\lvert I\rvert \lvert J\rvert)$
	RI	(YIP; CHEUNG; NG, 2004)	$(-\infty, \lvert J\rvert]$	Max	$O(N \lvert J\rvert)$
	C_r	(SANTAMARÍA; QUINTALES; THERÓN, 2007)	$[0,\infty)$	Min	$O(\lvert I\rvert^2 \lvert J\rvert)$
	C_c	(SANTAMARÍA; QUINTALES; THERÓN, 2007)	$[0,\infty)$	Min	$O(\lvert I\rvert \lvert J\rvert^2)$
	OC	(SANTAMARÍA; QUINTALES; THERÓN, 2007)	$[0,\infty)$	Min	$O(\max(\lvert I\rvert^2 \lvert J\rvert, \lvert I\rvert \lvert J\rvert^2))$
	MMSE	(CHEN; LIU; ZENG, 2015)	$[0,\infty)$	Min	$O(\min(\lvert I\rvert, \lvert J\rvert) \lvert I\rvert \lvert J\rvert)$
	SMSR	(MUKHOPADHYAY; MAULIK; BANDYOPADHYAY, 2009)	$[0,\infty)$	Min	$O(\lvert I\rvert \lvert J\rvert)$
Correlation-based	AC	(NEPOMUCENO; TRONCOSO; AGUILAR-RUIZ, 2011)	$[-1, 1]$	Max	$O(\lvert I\rvert^2 \lvert J\rvert)$
	SCS	(YANG; DAI; YAN, 2011)	$[0, 1]$	Min	$O(\max(\lvert I\rvert^2 \lvert J\rvert, \lvert I\rvert \lvert J\rvert^2))$
	ACV	(TENG; CHAN, 2008)	$[0, 1]$	Max	$O(\max(\lvert I\rvert^2 \lvert J\rvert, \lvert I\rvert \lvert J\rvert^2))$
	ASR	(AYADI; ELLOUMI; HAO, 2009)	$[-1, 1]$	Max	$O(\max(\lvert I\rvert^2 \lvert J\rvert \log \lvert J\rvert, \lvert J\rvert^2 \lvert I\rvert \log \lvert I\rvert))$
	SBM	(FLORES *et al.*, 2013)	$[0, \psi(B)\ \omega(B)]$	Max	$O(\max(\lvert I\rvert^2 \lvert J\rvert \log \lvert J\rvert, \lvert J\rvert^2 \lvert I\rvert \log \lvert I\rvert))$
Standardization-based	MSA	(GIRALDEZ *et al.*, 2007)	$[0,\infty)$	Min	$O(\lvert I\rvert \lvert J\rvert)$
	VE	(DIVINA *et al.*, 2012)	$[0,\infty)$	Min	$O(\lvert I\rvert \lvert J\rvert)$
	VEt	(PONTES; GIRÁLDEZ; AGUILAR-RUIZ, 2010)	$[0,\infty)$	Min	$O(\lvert I\rvert \lvert J\rvert)$

free implementations which are publicly available. These algorithms are based on different formulations and use diverse types of heuristics (e.g., greedy, divide-and-conquer, exhaustive enumeration, etc.) to deal with biclustering tasks. Hence, they are able to identify different types of bicluster patterns and bicluster structures (e.g., exclusive row or column biclusters, non-overlapping biclusters in checkerboard structures, arbitrarily positioned biclusters, etc.). Thus, they model different particularities of a dataset and reduce the bias towards a specific coherence measure when evaluating the identified biclusters. These algorithms are:

- *Cheng and Church's Algorithm (CCA)* (CHENG; CHURCH, 2000), which starts with the full data matrix as a bicluster. Next, it iteratively prunes rows and columns out of the bicluster, minimizing the MSR measure, until it satisfies a desired threshold. As a last step, some rows or columns are added back to the bicluster as long as they do not violate the MSR threshold.

- *Statistical-Algorithmic Method for Bicluster Analysis (SAMBA)* (TANAY; SHARAN; SHAMIR, 2002), which constructs a bipartite graph for the dataset, where one set of vertices represents the genes and the other set corresponds to the samples. Next, based on a likelihood model, it enumerates the most significant complete bipartite subgraphs (bicliques). Each biclique corresponds to a bicluster in the final solution.

- *Order Preserving Sub-Matrix (OPSM)* (BEN-DOR *et al.*, 2003), which mines biclusters containing columns that induce a permutation where the values of each row strictly increases. The search procedure is performed by a greedy heuristic guided by a probabilistic score.

- *Spectral* (KLUGER *et al.*, 2003) which searches for constant biclusters organized in a checkerboard structure. For such, it applies the singular value decomposition to the input matrix. Then, it clusters rows and columns independently by projecting them on their best partitioning eigenvectors and applying the k-means algorithm.

- *Plaid* (TURNER; BAILEY; KRZANOWSKI, 2005), which represents a set of biclusters as a sum of linear layers plus an additional layer that models noise and background effects in the data. The optimization problem consists of a sum of squared errors minimization between the plaid model and the data, which is solved by a binary least squares algorithm.

- *Binary Inclusion Maximal Biclustering Algorithm (Bimax)* (PRELIĆ *et al.*, 2006), which discretizes the input dataset into a binary matrix based on the threshold $\left(\min(A) + \max(A)\right)/2$, where $\min(A)$ and $\max(A)$ indicate the maximum and minimum values of the matrix. Next, it searches for upregulated biclusters whose values are all equal to one, using an enumerative divide-and-conquer approach.

- *Bayesian Biclustering (BBC)* (GU; LIU, 2008), which assumes the plaid model for the input dataset, but restricts the overlap between biclusters to occur only in genes or only in samples. For the plaid model fitting, it uses a Gibbs sampling procedure.

- *Large Average Submatrices (LAS)* (SHABALIN *et al.*, 2009), which assumes a Gaussian random matrix as a null model for the data and searches for biclusters with average values that significantly deviate from such a model. For such, it uses a greedy procedure to optimize a Bonferroni-based significance score that takes into account the size of a bicluster and its average value.

- *Qualitative Biclustering (QUBIC)* (LI *et al.*, 2009), which represents the data as a graph, with genes as vertices, edge weights equal to the number of samples for which two genes are similar. The algorithm consists of a greedy procedure that extracts biclusters that correspond to heavy subgraphs where the genes present similar expression patterns in the same subset of samples.

- *Factor Analysis for Bicluster Acquisition (FABIA)* (HOCHREITER *et al.*, 2010), which assumes a sum of multiplicative layers for a dataset, where each layer represents a different bicluster, plus a noise layer. To fit this model, FABIA uses an expectation-maximization approach for likelihood maximization.

In Table 3, we summarize the software packages used to implement the algorithms used in the experiments of this paper. The algorithms are available in R, Java, C and Python packages. In the experimental phase, we used biclustlib (PADILHA; CAMPELLO, 2017), which is a Python library that provides wrappers for these implementations.

Table 3 – Algorithms' software packages.

Algorithm	Language	Availability
CCA	R	<https://cran.r-project.org/web/packages/biclust/index.html>
SAMBA	Java	<http://acgt.cs.tau.ac.il/expander/>
OPSM	Java	<https://sop.tik.ee.ethz.ch/bicat/>
Spectral	Python	<https://scikit-learn.org/stable/>
Plaid	R	<https://cran.r-project.org/web/packages/biclust/index.html>
Bimax	R	<https://cran.r-project.org/web/packages/biclust/index.html>
BBC	C	<http://www.people.fas.harvard.edu/~junliu/BBC/>
LAS	Python	<https://github.com/padilha/biclustlib>
QUBIC	C	<https://github.com/maqin2001/qubic>
FABIA	Python	<https://github.com/bioinf-jku/pyfabia>

2.4.4 Data Collection

The experiments were performed using 19 datasets associated with the *Saccharomyces cerevisiae* organism, one of the organisms most comprehensively studied in biology and, as a consequence, with extensive and high-quality Gene Ontology information available (PRELIĆ *et*

al., 2006; CHRISTIE; HONG; CHERRY, 2009). This collection consists of the main biclustering benchmarks of this organism available in the literature. They are represented by dense real-valued data matrices obtained from time series microarray experiments, including the datasets used in (CHENG; CHURCH, 2000)[1] and (PRELIĆ *et al.*, 2006)[2], included in most biclustering studies, and the benchmark of 17 datasets introduced by (JASKOWIAK; CAMPELLO; COSTA, 2013)[3], whose data were systematically collected from previous gene expression data analyses studies (SPELLMAN *et al.*, 1998; CHU *et al.*, 1998; GASCH *et al.*, 2000) and were already used in clustering (JASKOWIAK; CAMPELLO; COSTA, 2014) and biclustering (PADILHA; CAMPELLO, 2017) analyses. The main aspects of these datasets are summarized in Table 4.

Table 4 – Gene expression datasets.

Name	# of genes	# of samples	Reference
Alpha factor	1099	18	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Cdc 15	1086	24	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Cdc 28	1044	17	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Elutriation	935	14	(JASKOWIAK; CAMPELLO; COSTA, 2013)
1mM menadione	1050	9	(JASKOWIAK; CAMPELLO; COSTA, 2013)
1M sorbitol	1030	7	(JASKOWIAK; CAMPELLO; COSTA, 2013)
1.5mM diamide	1038	8	(JASKOWIAK; CAMPELLO; COSTA, 2013)
2.5mM DTT	991	8	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Constant 32nM H2O2	976	10	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Diauxic shift	1016	7	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Complete DTT	962	7	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Heat shock 1	988	8	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Heat shock 2	999	7	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Nitrogen depletion	1011	10	(JASKOWIAK; CAMPELLO; COSTA, 2013)
YPD 1	1011	12	(JASKOWIAK; CAMPELLO; COSTA, 2013)
YPD 2	1022	10	(JASKOWIAK; CAMPELLO; COSTA, 2013)
Yeast sporulation	1171	7	(JASKOWIAK; CAMPELLO; COSTA, 2013)
S. cerevisiae	2993	173	(PRELIĆ *et al.*, 2006)
Tavazoie	2884	17	(CHENG; CHURCH, 2000)

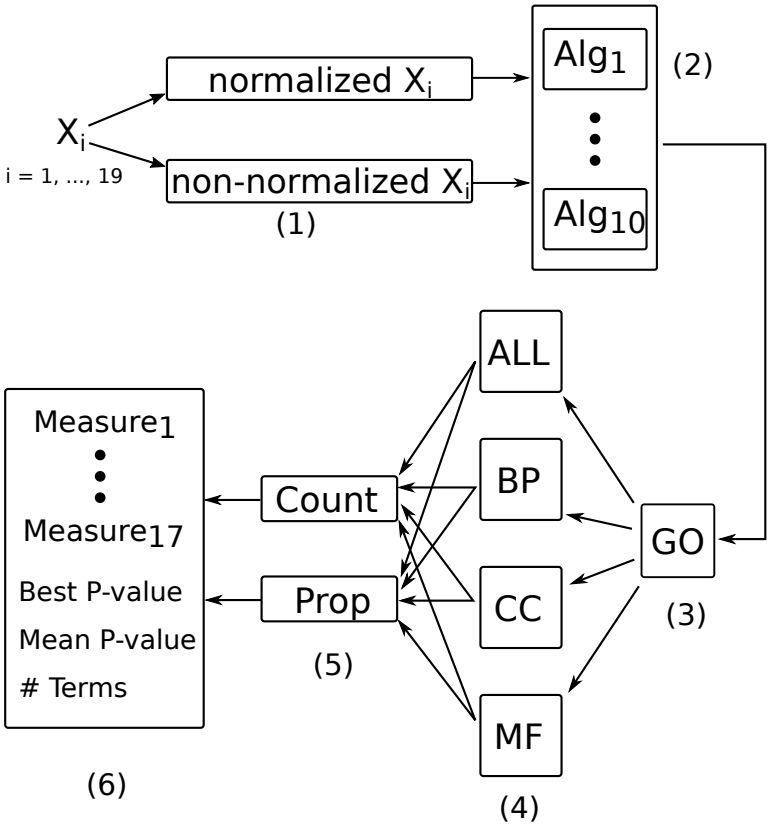
2.4.5 External Bicluster Evaluation Measures

For the external evaluation, we performed the gene enrichment analysis of the biclusters found using the Gene Ontology (GO)[4] (ASHBURNER *et al.*, 2000) knowledge base, which provides three ontologies: Biological Process, Molecular Function and Cellular Component. Each ontology contains a structured general vocabulary comprising "is-a" and "part-of" relationships between its terms to describe the role of the genes in an organism (CONSORTIUM, 2004).

In this study, we performed four different analyses using the GO database: (i) using all the three ontologies; (ii) using only the Biological Process ontology; (iii) using only the Cellular Component ontology; and (iv) using only the Molecular Function ontology. For each

[1] <http://arep.med.harvard.edu/biclustering/>
[2] <https://sop.tik.ee.ethz.ch/bimax/>
[3] <http://lapad-web.icmc.usp.br/repositories/ieee-tcbb-2013/index.html>
[4] <http://www.geneontology.org/>

Figure 5 – Experimental methodology followed to obtain the results of the coherence and GO measures.



analysis, after identifying the GO terms in each bicluster, the Fisher test was applied to assess the over-representation of each term (TANAY; SHARAN; SHAMIR, 2002; PRELIĆ *et al.*, 2006; LI *et al.*, 2009; PADILHA; CAMPELLO, 2017). In this study, a GO term was considered significant in a bicluster if its p-value, after performing the Benjamini and Hochberg multiple test correction (HOCHBERG; BENJAMINI, 1990), was lower than 0.05 (EREN *et al.*, 2012; PADILHA; CAMPELLO, 2017). Three different measures were extracted for each bicluster containing at least one significant term (PONTES; GIRLDEZ; AGUILAR-RUIZ, 2015): the mean p-value, the best p-value and the number of significant terms. The experiments investigated correlations of these quantities with the coherence measures discussed in Section 2.4.2.

2.4.6 Experimental methodology

Briefly, the experimental methodology has 6 steps, which are illustrated in Figure 5. We will now explain each step.

Given a dataset X_i, two different scenarios were considered in step (1) before applying any algorithm and coherence measure. In the first scenario, the features (samples) of each dataset were standardized to zero mean and unit variance. In the latter scenario, the algorithms and coherence measures were applied to the original (non-normalized) data.

In step (2), the selected algorithms were run on both versions of X_i. Deterministic and

non deterministic algorithms were selected for this study. For each dataset, the deterministic algorithms (SAMBA, OPSM, Bimax and QUBIC) were run once, while the non deterministic algorithms (CCA, Spectral, Plaid, BBC, LAS and FABIA) were run 30 times.

In step (3), the biclusterings found by each algorithm were compared with the GO external evaluation. Four different scenarios were considered for the GO evaluation, which are illustrated in step (4): (i) using all GO ontologies (ALL); (ii) using only the Biological Process ontology (BP); (iii) using only the Cellular Component ontology (CC); and (iv) using only the Molecular Function ontology (MF).

Given that 6 out of the 10 investigated algorithms are non deterministic, a pre-established procedure was adopted to select which of their biclusterings would be analyzed for each dataset. In step (5), two different approaches were followed. The first, called "count", selects, for each dataset, the biclustering solution that contains the median number of significant biclusters. The second, called "prop", selects, for each dataset, the biclustering solution that contains the median proportion of significant biclusters for the total number of biclusters in the solution[5]. Both approaches do not discard empty biclustering solutions to calculate the median.

Finally, in step (6), the 17 coherence measures from Section 2.4.2 and the 3 GO measures from Section 2.4.5 are calculated for each bicluster containing at least one significant GO term, according to the GO scenario being considered.

2.4.7 Hyperparameter values used for the algorithms

The hyperparameter values used in this study were usually based on the default settings used or recommended by the original authors of each algorithm. However, to achieve results that best fit the investigated scenarios, they were modified for some of the biclustering techniques. These modifications are explained next.

CCA requires a maximum MSR threshold δ to produce biclusters. This quantity is usually different from one dataset to another. In this paper, $\delta = (\max(A) - \min(A))^2/12 \times 0.005$ (HORTA; CAMPELLO, 2014), where $\max(A)$ and $\min(A)$ indicate the maximum and minimum values of a dataset, respectively. This setting provides an approximation for the δ values considered in the original work of Cheng and Church (2000).

Before running its Gibbs sampling procedure, BBC normalizes the dataset. The interquartile range normalization (IQRN) on the features proposed by its original authors was not used here. Instead, we used the zero mean and unit variance normalization for the scenario of normalized data, to be in accordance with the other algorithms used in this study.

For the number of biclusters, 7 algorithms (CCA, Plaid, Bimax, BBC, LAS, QUBIC and FABIA) were executed to search for 30 biclusters in each dataset. Spectral was run to search

[5] This approach is different than "count" because it is not guaranteed that the heuristic adopted by each algorithm will always return the same number of biclusters.

for 15 gene clusters and 2 sample clusters. The other algorithms (SAMBA and OPSM) do not receive the number of biclusters as a hyperparameter. Thus, all biclusters returned by them were considered.

2.5 Results and discussion

Overall, we evaluated 16 different experimental scenarios, by combining: 2 versions of the datasets (normalized and non-normalized), 4 ontology analyses (ALL, BP, CC and MF), and 2 approaches to select biclusterings generated by non deterministic algorithms ("count" and "prop"). For each scenario, the biclusters found by all algorithms in all datasets were initially concatenated in an array. Next, the Pearson and Spearman correlations were calculated for the previously discussed coherence and external measures. The results are illustrated as heatmaps in Figures 6 and 7, where each element corresponds to the correlation value. To save space, only the correlations with the "count" approach, normalized data, and the three GO ontologies (ALL) are shown. The other 15 scenarios achieved similar results in most cases, allowing us to draw similar conclusions. Their respective figures are available in our supplementary material[6]. Minor differences are discussed in the text.

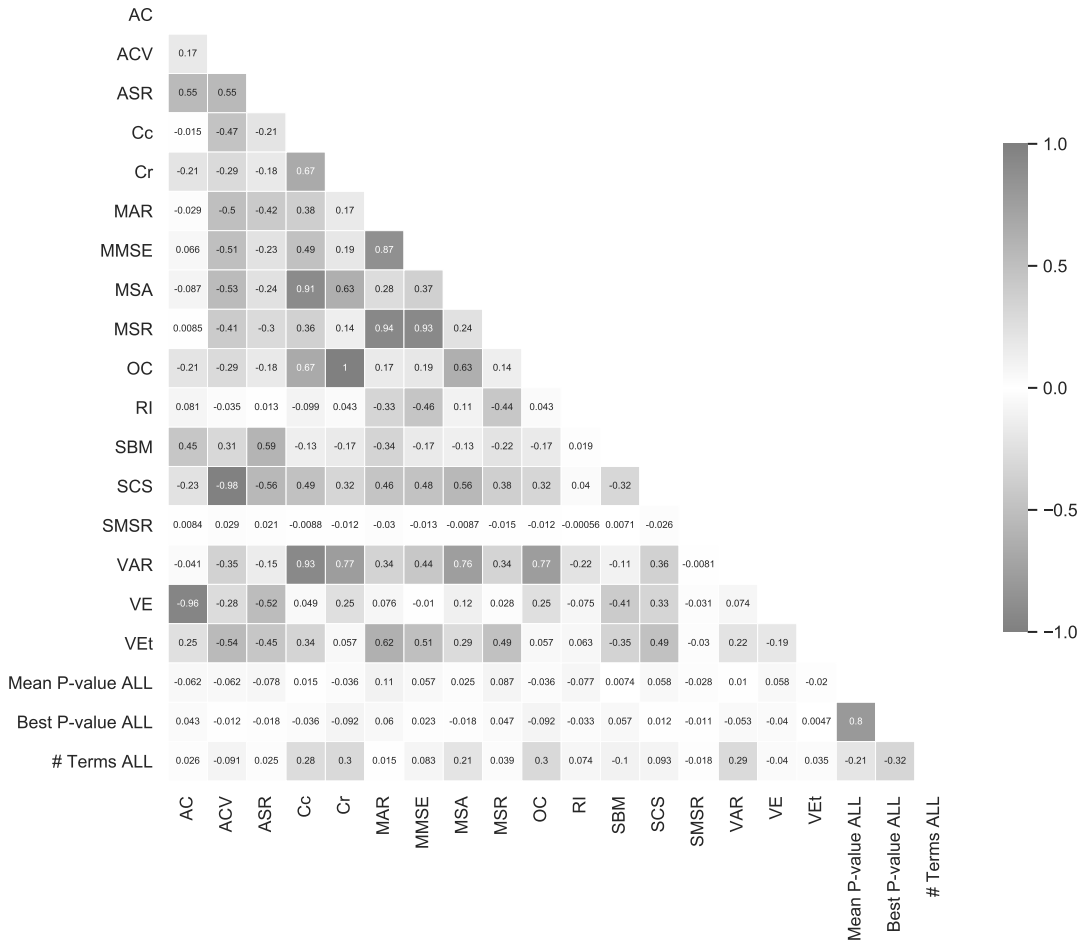
According to Figures 6 and 7, the measures from the external evaluation are not strongly correlated with any coherence criterion. These results were observed for all investigated scenarios. Therefore, biclusters with high biological significance from the GO point of view do not necessarily imply in good values for the coherence measures. Thus, it may be feasible to recommend using multiple bicluster coherence criteria to complement the GO analysis. As a result, the biclusters will also be evaluated by a set of predefined patterns of interest and one can carefully inspect the quality of their trends.

It can be seen that some coherence criteria presented similar behavior according to the correlations. Measures that must be either maximized or minimized were selected. Thus, the interest is in strong correlations that can be either positive or negative. From the results, a few pairs of strongly correlated measures, with a correlation above 0.9 or below -0.9, can be extracted:

- (OC, C_r), (SCS, ACV) and (VE, AC) for both correlation coefficients in all experimental scenarios;
- (MSR, MAR) for Spearman in all scenarios and for Pearson in all scenarios with normalized data and in the scenario with non-normalized data and MF analysis;
- (VAR, C_r) and (VAR, OC) for the Pearson coefficient in all scenarios using non-normalized data;

[6] <http://padilha.github.io/asoc-2019-suppl>

Figure 6 – Results of the Pearson correlation using normalized data, all ontologies and the "count" approach.



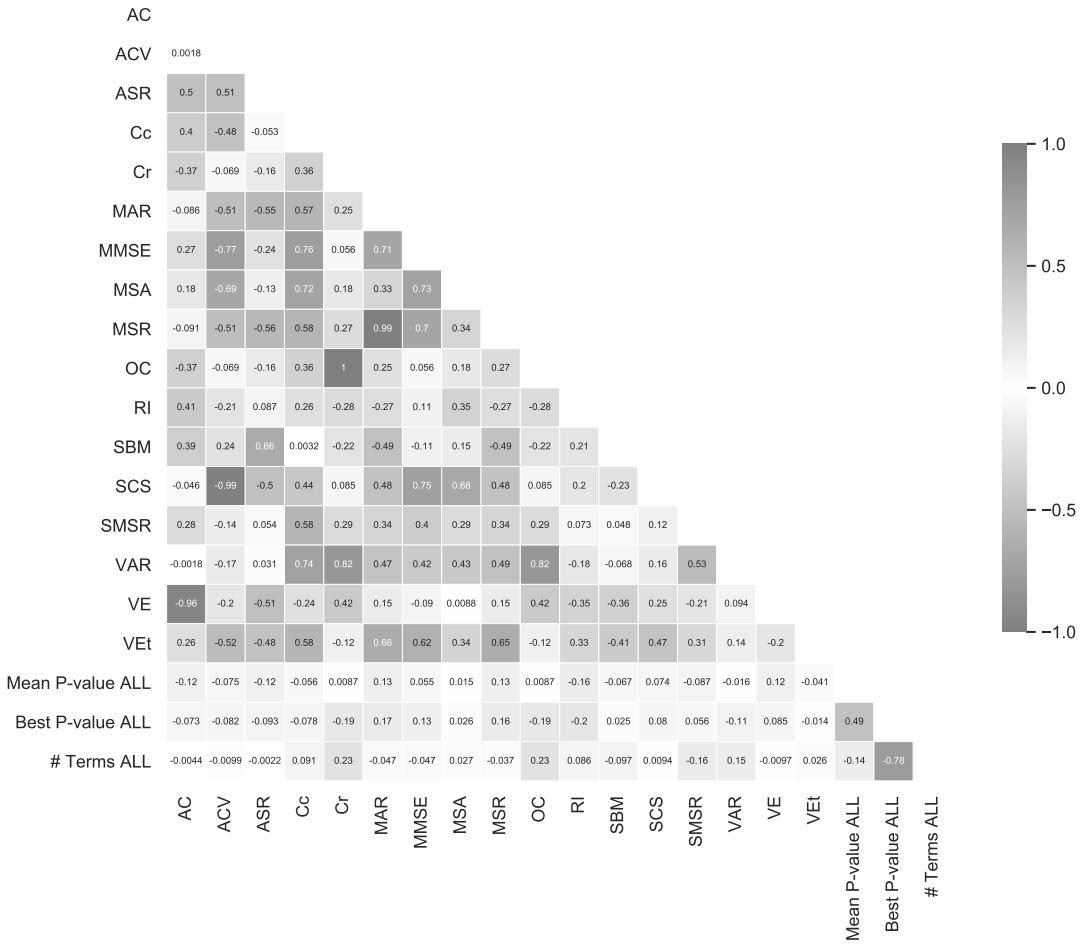
- (VAR, C_c) and (MSA, C_c) for the Pearson coefficient in all scenarios with normalized data; and
- (MSR, MMSE) for the Pearson correlation in all experimental scenarios.

It can be observed that the strong correlated pairs contain measures that detect similar patterns. To avoid using paired criteria in the same application, since their results will be redundant, the one that is able to detect the most general numeric patterns is recommended.

Some evidence can be found that data normalization may be determinant in the behavior of some pairs of measures. This result was expected, since the algorithms do not return the same biclustering solutions when dealing with non-normalized or normalized data. Moreover, normalized data may alleviate the influence of different feature scales or outliers in the behaviors of the measures.

In addition, the correlations between measures might be different when considering different ontology scenarios for the enrichment, as was observed for (MSR, MAR) and the Pearson correlation.

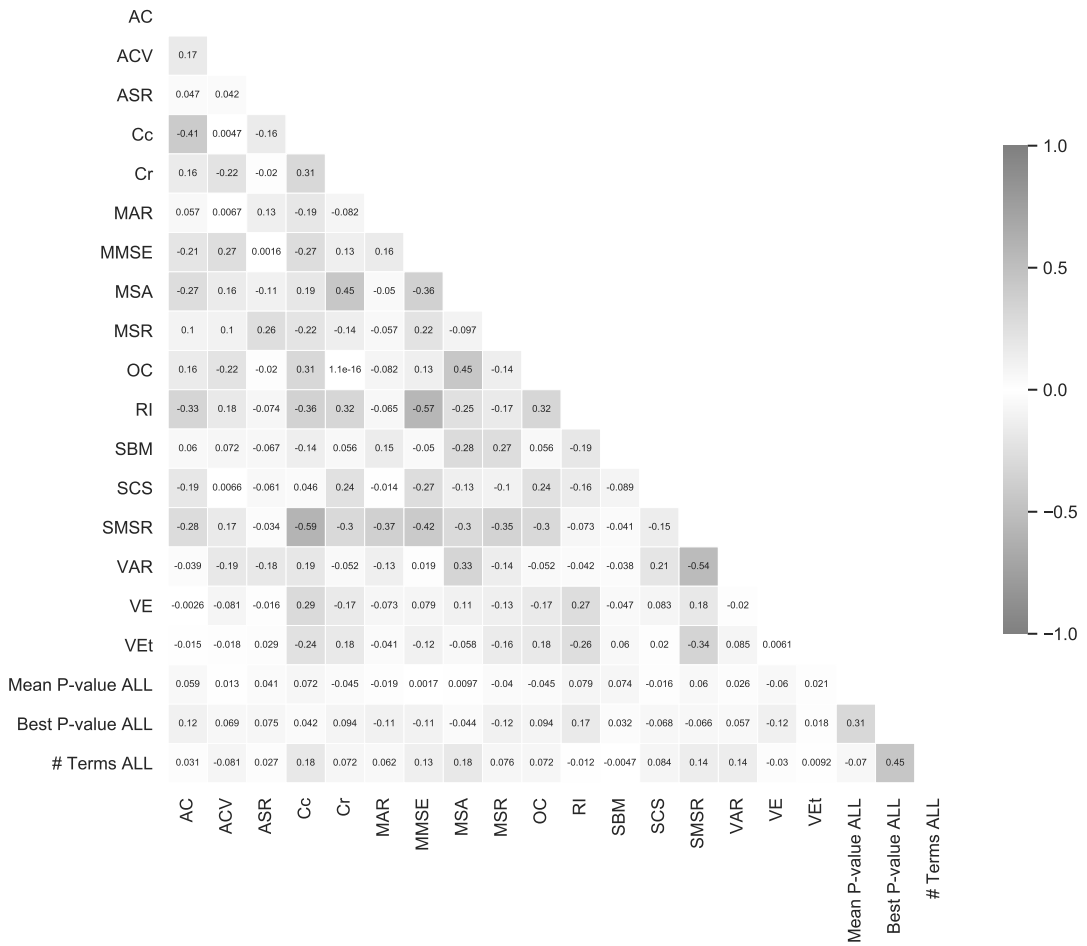
Figure 7 – Results of the Spearman correlation using normalized data, all ontologies and the "count" approach.



Real applications may benefit from favouring measures with the lowest computational complexities. Table 2 summarizes the investigated measures and their computational complexities. Even if two coherence measures present lower correlations (e.g., around 0.7 or 0.8), those with lower complexity should be preferred, especially if a large number of biclusters is evaluated. From this table, the measures with the lowest complexities are: VAR, MSR, MAR, SMSR, MSA, VE and VEt.

The difference between the results from the Pearson and Spearman correlations, shown in Figure 8, were also analyzed. The difference observed was low for many pairs, which indicates that the two correlations were compatible in most cases. To statistically validate this finding, the Wilcoxon signed-rank test was applied to the difference matrix. Under a significance level of 0.05 no statistical evidence of difference was found, which supports the agreement of the correlation matrices. We repeated the Wilcoxon signed-rank test on each of the other 15 experimental scenarios. In all cases, we did not find statistical evidence to reject the null hypothesis.

Figure 8 – Difference between Pearson and Spearman correlations.



2.6 Conclusions

This paper extended the work of Pontes, Girldez and Aguilar-Ruiz (2015) by investigating the behavior of 17 bicluster coherence measures. We applied them to the results of 10 well-established biclustering algorithms. Our experiments were performed on a benchmark of 19 *Saccharomyces cerevisiae* time-course datasets.

The correlations among the coherence and the external GO criteria were analyzed using the Pearson and Spearman coefficients. According to the analysis, external GO evaluations did not agree with any coherence measure. These results suggest that a high GO significance does not automatically imply in good evaluations with coherence criteria. Besides, GO information may be incomplete (SANTAMARÍA; QUINTALES; THERÓN, 2007). Thus, the use of bicluster coherence measures together with the GO analysis may be a better alternative to achieve more concrete conclusions.

These results conflict with those from Pontes, Girldez and Aguilar-Ruiz (2015), which claimed that the coherence measures present some dependence with the external biological measures. However, since this study employed 10 different algorithms, it reduced the bias

regarding the evolutionary algorithm used in Pontes, Girldez and Aguilar-Ruiz (2015).

Overall, we analyzed 16 different experimental scenarios, which included: normalized and non-normalized data, evaluation using all GO ontologies, and 2 different approaches to select the results of non deterministic algorithms. We observed that normalization and the GO validation approach may be determinant, since some pairs of measures presented strong Pearson correlations in scenarios using either normalized or non-normalized data and specific ontologies for the enrichment.

In practical applications, the users of the measures must take into account the types of correlations among measures that they want to avoid. For such, we advise them to consider as similar only the pairs that presented a strong correlation in all scenarios for the desired coefficient (Pearson or Spearman) and data type (normalized or non-normalized).

This study also presented the time complexity analyses of the measures, usually not provided in their original studies. In many applications, the time complexities may be an important reason for choosing some measures rather than others. Mainly when a large number of biclusters need to be evaluated and/or the biclusters may be constituted by a large number of rows and columns, measures with the lowest complexities may be preferred.

Finally, the choice of the most appropriate bicluster coherence measure must also take into account the task to be solved. In a few practical scenarios, one may favor particular types of patterns compared to others and/or may prioritize measures with lower computational complexities. However, the use of heterogeneous measures allows the analysis of biclusters with different points of view. According to the experimental results reported in this paper, it is possible to avoid selecting a set of measures that present redundant behavior and may not bring new insights to the analysis.

CHAPTER

3

CRISPRCASIDENTIFIER: MACHINE LEARNING FOR ACCURATE IDENTIFICATION AND CLASSIFICATION OF CRISPR-CAS SYSTEMS

Publication information: This chapter is an article that was published in the GigaScience journal by Oxford University Press. The article is licensed under CC BY 4.0*.

Reference: PADILHA, V. A.†; ALKHNBASHI, O. S.†; SHAH, S. A.; DE CARVALHO, A. C. P. L. F.; BACKOFEN, R. CRISPRcasIdentifier: Machine learning for accurate identification and classification of CRISPR-Cas systems. **GigaScience**, v. 9, n. 6, 06 2020. ISSN 2047-217X. Giaa062. Available: <https://doi.org/10.1093/gigascience/giaa062>.

3.1 Abstract

CRISPR-Cas genes are extraordinarily diverse and evolve rapidly when compared to other prokaryotic genes. With the rapid increase in newly sequenced archaeal and bacterial genomes, manual identification of CRISPR-Cas systems is no longer viable. Thus, an automated approach is required for advancing our understanding of the evolution and diversity of these systems, and for finding new candidates for genome engineering in eukaryotic models. We introduce CRISPRcasIdentifier, a new machine learning based tool that combines regression and classification models for the prediction of potentially missing proteins in instances of CRISPR-Cas systems and the prediction of their respective subtypes. In contrast to other available tools, CRISPRcasIdentifier can both detect *cas* genes and extract potential association rules that reveal functional modules for CRISPR-Cas systems. In our experimental benchmark on the

*<https://creativecommons.org/licenses/by/4.0/>

† Contributed equally as first authors.

most recently published and comprehensive CRISPR-Cas system dataset, CRISPRcasIdentifier was compared with recent and state-of-the-art tools. According to the experimental results, CRISPRcasIdentifier presented the best Cas protein identification and subtype classification performance. Overall, our tool greatly extends the classification of CRISPR cassettes and, for the first time, predicts missing Cas proteins and association rules between Cas proteins. Additionally, we investigated the properties of CRISPR subtypes. The proposed tool relies not only on the knowledge of manual CRISPR annotation but also on models trained using machine learning.

3.2 Introduction

CRISPR-Cas systems provide archaea and bacteria with a nucleic acid based adaptive immune system against invading viruses and plasmids. Mechanistically, the immune response can be divided into three stages, namely adaptation, processing and interference, each carried out by different sets of protein complexes (GARNEAU, 2010). The universally conserved proteins Cas1, Cas2 and, optionally Cas4, are responsible for the adaptation stage, when a fragment of invader DNA is excised and stored in the host chromosome as a spacer in the non-coding CRISPR region. The processing and interference stages are much more mechanistically diverse, using different sets of proteins, depending on the type of CRISPR-Cas system. CRISPR-Cas systems are found in many bacteria and most archaea, and have diversified as much as their host organisms (MAKAROVA *et al.*, 2015).

While the mechanistic principles are similar, with spacers comprising templates for synthesis of CRISPR interference RNAs (crRNAs) against the invader, the different types and classes of CRISPR-Cas systems show some important differences. Class 2 systems use a single multi-domain protein for locating and cleaving the re-invading nucleic acid, whereas Class 1 systems employ a large multi-subunit complex for the same purpose. Class 2 systems can be further subdivided into type II, V and VI, which appear to have evolved independently from each other. Thus, the respective Cas9, 12 and 13 enzymes that carry out invader cleavage rely on diverse mechanisms involving differing nuclease domains (MAKAROVA *et al.*, 2015; SHMAKOV *et al.*, 2015; SHMAKOV *et al.*, 2017).

Class 1 systems, on the other hand, with types I, III and IV, use structurally related proteins to carry out similar functions, although the protein subunits have diverged considerably. Common to all Class 1 systems is that Cas7 forms a helical backbone that spans the length of the tightly bound crRNA. This backbone is terminated in one end by Cas5, which itself is bound to Cas8 or Cas10 for type I and IV, or type III systems, respectively. Type I systems use Cas8 for the recognition of the protospacer adjacent motif (PAM) (CASS *et al.*, 2015), which, along with invader crRNA hybridisation, comprises a signal for recruitment of the Cas3 helicase-nuclease protein that subsequently digests the invader chromosome (SINKUNAS *et al.*, 2011). Type III systems, however, use the Cas7 backbone for cleaving invader mRNA while the

Cas10 HD nuclease cleaves transcribed DNA (ZHANG *et al.*, 2012; DENG *et al.*, 2012). Cas10 also synthesises a signaling molecule that recruits additional accessory Cas proteins for other functions, such as cell suicide or activation of other defense systems (DENG *et al.*, 2012; SHAH *et al.*, 2018).

The different types of CRISPR-Cas systems are themselves so diverse that each type can be further subdivided into several subtypes. Type III, for example, is divided into four subtypes III-A, B, C and D. While CRISPR-Cas systems of the same subtype encode similar proteins that occupy the same roles, the proteins have often diverged beyond the point of recognition by conventional sequence alignment methods such as BLAST, even within a subtype. This level of sequence diversity makes proper identification of the found CRISPR-Cas systems very challenging, and the field has thus far relied upon the gold standard of periodic manual annotations by experts, published once every few years (HAFT *et al.*, 2005; MAKAROVA *et al.*, 2011; MAKAROVA *et al.*, 2015). The annotation involves profile HMM searches for finding core genes, followed by the inspection of their neighborhoods, gauging operonic structures, and manual BLAST and PSI-BLAST searches (MARCHLER-BAUER; BRYANT, 2004). With the increasing number of genome sequences from uncultured microbes and metagenomic data, however, manual annotations cannot keep up and an automated approach is needed, which yields comparable accuracy to manual annotation. Furthermore, research groups working on organisms not yet covered by published annotations have thus far made their own manual annotations, leading to inconsistencies in nomenclature and inaccuracies in some cases.

There have been numerous attempts at devising computational pipelines for the identification of different elements of the CRISPR system, such as CRISPR arrays (LANGE *et al.*, 2013; ALKHNBASHI *et al.*, 2014; BISWAS *et al.*, 2016) and CRISPR leaders (ALKHNBASHI *et al.*, 2016). On the other hand, command line tools and webservers, usually based on Hidden Markov Models (HMM) and HMMER (FINN; CLEMENTS; EDDY, 2011) or PSI-BLAST (MARCHLER-BAUER; BRYANT, 2004), were proposed for CRISPR subtype prediction. Examples of such tools are: CRISPRdisco (CRAWLEY *et al.*, 2018), CRISPRcasFinder (COUVIN *et al.*, 2018), Macsyfinder (ABBY *et al.*, 2014), CRISPRone (ZHANG; YE, 2017) and HmmCas (CHAI *et al.*, 2019). We found, however, that the existing tools usually lack the ability to generalize unseen examples. Additionally, these tools are neither able to adapt to an extending repertoire of *cas* genes, nor predict possibly missing proteins, nor can learn association rules among proteins.

In this work, we present a machine learning (ML) approach intending to capture much of the relevant essence of manual annotation. It is based on evidence for the different Cas proteins to be contained in a series of consecutive genes that are part of a cas cassette, and thus represents genomic CRISPR-Cas systems as cassettes of adjacently encoded proteins. These evidences are calculated by newly designed sets of HMM models for each Cas protein, covering the diversity of Cas protein families. The proposed approach solves the problem of classification of new systems

into types and subtypes. As our features for the ML approach correspond to evidence for Cas proteins, we can determine Cas proteins whose evidence is critical for predicting a subtype, which corresponds to the concept of signature genes. We show that our approach correctly identifies known signature genes for types and subtypes. In addition, our approach is able to provide more information about the composition of cassettes. One application is to predict evidence for Cas proteins that have been missed in the Cas protein screening. This provides researchers with hints to search for remote homologs of the missing Cas proteins, or for new proteins that might replace the associated function. Furthermore, we are able to learn association rules, which are subsets of proteins being important to each other, indicating functional modules. As a proof of concept, when we search for Cas proteins associated with an interference protein, our approach finds other interference proteins to be most important. The more interesting cases are undoubtedly with non-interference proteins, where our tool could correctly predict a strong association of the ancillary protein Csn2 with Cas1, consistent with its hypothesized role in adaptation. For the protein CasR we found that it is associated with different functional modules in subtypes I-A and I-E, indicating a possible functional diversity. Thus, the set of protein associations derived in this manner provides a proper resource for researchers that want to investigate the function of different Cas proteins.

3.3 Methods

3.3.1 Data collection and preprocessing

All Cas proteins used in this study were selected from the current classified archaeal and bacterial CRISPR-Cas systems (MAKAROVA *et al.*, 2015; SHMAKOV *et al.*, 2015; SHMAKOV *et al.*, 2017). We performed an all-against-all sequence similarity comparison on these data using Fasta (PEARSON; LIPMAN, 1988). Subsequently, we clustered the proteins using the Markov Cluster Algorithm (MCL) (ENRIGHT *et al.*, 2002) based on custom similarity criteria (ALKHNBASHI *et al.*, 2016; SHAH *et al.*, 2018). These criteria consider the size of the proteins, the length alignment and the relative locations of similar regions between the two compared proteins. After clustering the protein sequences from a specific Cas protein family, we generated a multiple sequence alignment (MSA) using MUSCLE (EDGAR, 2004). Next, these alignments were converted to HMM profile models by using hmmbuild (FINN; CLEMENTS; EDDY, 2011). Except for MCL, all other tools were run with default parameters.

Throughout the text, each cassette in our training and test data sets is represented by a tuple consisting of its genomic sequence containing all genes of the cassette, and the list of all annotated Cas proteins. We extracted the genomic sequences as follows: we took the Supplementary Table S7 from (MAKAROVA *et al.*, 2015), which contains all gene loci (i.e. genomic positions) in column "(sub)Type / Coordinates", and downloaded the sequences from NCBI. For the second part, namely the list of all Cas proteins, we extracted the genomic sequence

for each annotated Cas protein individually, again from the "coordinates" column and added 50 bases of context. The associated amino-acid sequences were generated by running the Prodigal tool v2.6.3 (HYATT *et al.*, 2010) on the respective gene sequences, and stored together with the Cas annotation from the column "*cas* gene" in Table S7 from (MAKAROVA *et al.*, 2015).

To generate the feature vectors, we ran all HMM profile models using hmmsearch against the sequences of all cassettes. We selected the cassettes, which had a hit for all proteins annotated for that subtype, and used this as training and test set for the classification pipeline. Cassettes that had a missing protein were used instead as an independent test case for our regression models and the full pipeline.

3.3.2 Classification of Cas cassettes

For this task, we apply ML algorithms onto a finite sample of CRISPR data in order to obtain predictive models that are able to classify Cas cassettes into their respective subtypes using a data matrix representation (see Results and discussion). Thus, based on the finite sample of data, we investigate the application of classification algorithms that estimate a function which is able to generalize the association between a cassette and its subtype. As a consequence, we intend to use this function to classify new cassettes that were not seen during the training phase into their respective subtypes with a high level of accuracy.

3.3.3 Prediction of missing Cas proteins

We also investigate the problem of predicting (possibly) missing Cas proteins by estimating their normalized bit scores. For this problem, we modelled it as follows. Given m Cas proteins, we filter, for each subtype, its set of $l < m$ proteins (i.e., all Cas proteins whose bit score is larger than zero for at least one cassette of the subtype). Next, we train l regressors, where the j^{th} regressor, $j \in \{1, \cdots, l\}$, predicts the bit score of the j^{th} Cas protein using the remaining $l-1$ proteins as input.

3.3.4 Experimental evaluation of ML algorithms

Three ML algorithms were applied to the preprocessed dataset to train classification and regression models:

- *Classification and Regression Trees (CART)* (BREIMAN *et al.*, 1984), which trains a predictive model represented by a decision tree. This algorithm can train decision trees for classification (classification trees) and regression (regression trees) tasks. A decision tree is composed by a set of interpretable rules extracted from the training dataset. These rules explain the decisions made by the model to predict the class or regression value for new, previously unseen, examples.

- *Support Vector Machines (SVM)* (VAPNIK, 1995), which trains a binary classifier represented by a hyperplane that separates examples from two classes with the maximum possible separation margin. By using kernel functions, an SVM can be applied to non-linearly separable problems. For multiclass classification tasks, a multiclass dataset is usually first decomposed into several classification binary datasets. SVMs can then be applied to each binary dataset, and their predictions are combined for a multiclasss classification.

- *Extremely Randomized Trees (ERT)* (GEURTS *et al.*, 2006), uses an ensemble of decision trees, where each tree is trained using a random subset of the original features. Instead of selecting the best discriminating threshold for each feature considered for a split, as would be the case for classical decision trees, ERT chooses a random threshold value. The final predictions are the average of the predictions of all the decision trees in the ensemble. We can extract the importance of each feature in the classification or regression task from the decision trees in the ensemble. The importance is represented by the decrease in impurity caused by a node that splits the feature, weighted by the number of examples contained in such a node (WU *et al.*, 2008), and averaged over all trees of the ensemble.

The model selection and evaluation of predictive models is a widely studied problem in the ML literature. Several works, such as Varma and Simon (2006), Cawley and Talbot (2010), Krstajic *et al.* (2014), investigate the advantages and drawbacks of different methodologies. Based on these previous studies, we use the nested cross-validation procedure. Given a set of data, the classical cross-validation approach splits the data into K mutually exclusive and similar sized subsets called folds. Next, at each iteration, $K-1$ folds are used for training an ML model and the remaining fold for testing it (BISHOP, 2006; HASTIE; TIBSHIRANI; FRIEDMAN, 2009). The nested cross-validation approach separates the model selection and evaluation steps, by using two different cross-validation loops: an outer loop, which splits the data into K_1 folds, and is used for model evaluation; and an inner loop, which splits the training data into K_2 folds, and is used for model selection. In this paper, we set $K_1 = K_2 = 10$, and repeat the evaluation procedure 50 times, due to the variance of the results when considering different splits (KRSTAJIC *et al.*, 2014). It is important to mention that, during our experiments, to guarantee that examples from all classes are present in each outer fold, we used only classes containing at least 10 examples.

For each cross-validation iteration, we aggregate the predictions from all folds and calculate a single predictive performance evaluation, in order to avoid any averaging problems that might arise, especially when the dataset is imbalanced (FORMAN; SCHOLZ, 2010). For the classification experiments, we used the following evaluation measures: adjusted balanced accuracy score (BRODERSEN *et al.*, 2010; GUYON *et al.*, 2015), an adaptation of the original accuracy measure that gives higher weights to examples from smaller classes; and the F-score with macro-averaging (SOKOLOVA; LAPALME, 2009), which is the average F-score among all classes. Both measures treat different subtypes equally. Thus, they do not favor those with

the largest numbers of cassettes. For the regression experiments, we used the mean absolute error (WILLMOTT; MATSUURA, 2005), which is the average absolute difference between the expected and the predicted target values.

Regarding the model selection step of each ML algorithm used, we performed a grid search over 20 different hyperparameter combinations, based on the guidelines from the scikit-learn package (PEDREGOSA *et al.*, 2011). We describe these hyperparameter grids next. For the CART algorithm, we varied the hyperparameters that determine the maximum depth of the decision tree and the minimum number of examples necessary for a node to become a leaf. For the former, we considered the values in $\{5, 10, 15, \max\}$, where max allows the tree to grow as deep as possible. For the latter, we varied the values in $\{5, 6, 7, 8, 9\}$. For the SVM algorithm, we used a Gaussian kernel, due to its ability to model nonlinear decision boundaries and its reduced number of hyperparameters when compared with another commonly used nonlinear kernel, the polynomial kernel (HSU *et al.*, 2003). For the cost hyperparameter C, we considered the values in $\{1, 10, 100, 1000\}$. Regarding the kernel coefficient γ, we assessed the values in $\{0.01, 0.1, 1, 10, 100\}$. Finally, for the ERT algorithm, we varied the ensemble size using the values in $\{25, 50, 75, 100\}$, and the quantity of features to be considered when performing a split from the set of values in $\{25\%, 50\%, 75\%, 100\%, \sqrt{m}\}$, where m is the number of known Cas protein families.

3.4 Results and discussion

3.4.1 A combined approach to determine Cas proteins and cassette subtypes

The classification of a subtype is based on the membership for specific Cas proteins. Thus, any ML-based classification of a cassette requires the detection of the contained Cas proteins as a first step. While this first step is commonly performed using Hidden Markov models, a difficulty arises from the fact that a single Cas protein family has to be split into different subfamilies due to the high evolutionary diversity of their members. Due to missing values in the dataset for a family, even the problem of splitting into different subfamilies is not an easy one. Even further, we have observed that the splitting of Cas protein families influence the quality of ML-based subtype classification. This would be quite obvious if subfamilies of individual Cas protein would correlate well with subtypes. The real situation, however, is more complex, partially due to the fact that cassettes are composed in a modular way, often involving horizontal gene transfer (MAKAROVA *et al.*, 2015; SHAH *et al.*, 2018).

In brief and as described in more detail later, our classification approach takes the bit scores for the contained Cas proteins as evidence of their membership to the cassette. We use this information to apply a set of ML algorithms to classify the subtypes of cassettes. By generating different divisions of subfamilies for each Cas protein, we obtain different evidences

for the contained Cas proteins. Thus, we can investigate which division is best related to subtype evolution. With this holistic view of Cas protein and subtype annotation, we can further examine relations between subtypes and Cas protein membership and as a result reassure key components of subtypes like signature genes.

3.4.2 Detection of Cas proteins by families of HMMer models

Our definition of Cas protein subfamilies is based on clustering the known sequences of a specific Cas protein family. We use around 68594 Cas proteins as a database, and applied different cluster criterion. Each cluster characterizes a subfamily, which is afterwards represented by a HMM model. All models for a Cas protein are grouped, and the best matching HMM for each Cas protein is used to score a new sequence. To cluster the sequences, we performed an all-against-all sequence similarity comparison. Subsequently, we applied the Markov Cluster Algorithm (MCL) (ENRIGHT *et al.*, 2002) to cluster the known sequences for a specific Cas protein family according to their sequence similarities. However, protein sequences can be clustered in different ways, depending on the cut-off for sequence similarity and the requested coverage of the alignment between two sequences. In addition, different hyperparameters for the MCL clustering algorithm result in different data partitions. Each partition defines different subfamilies, for which we train HMM models.

The different clustering approaches thus result in HMM models for different subfamilies, with varying specificity and sensitivity to detect members of a Cas protein family. We created five different collections of HMM models labeled HMM_1 ... HMM_5 using different hyperparameter values for the clustering algorithm and distinct threshold values for the all-against-all sequence similarity detection (see Methods for detail; the number of models for each Cas protein family is listed in Supplementary Table 11). For a given Cas protein sequence, we applied all HMM models that are contained in a specific collection for that protein family and took the maximum bit score, and zero otherwise. Non-zero values indicate that the investigated protein sequence belongs to the Cas protein family defined by the HMMer model set.

We used different measurements to assess the quality of a specific division represented by a set HMM_i. One quality criteria for a set HMM_i is clearly the capability for detecting known members of Cas proteins. Table 5 shows the sensitivity for the five sets HMM_1 ... HMM_5 by reporting the number of cassettes found in each subtype. It is easy to see that the more fine grained sets, HMM_1, HMM_2 and HMM_3, clearly detect more Cas proteins than the less fine grained sets HMM_4 and HMM_5.

In our holistic view of Cas protein detection and subtype classification, however, we want to understand also how the division into subfamilies relates to the cassette subtype, and thus influence the subtype classification. For that reason, we show in Table 5 also as another quality criteria the median accuracy for correctly predicting the subtype of a cassette when using the HMM_i in a ML-based subtype classification approach as described in the next section. The

Table 5 – Properties and Quality Measurements for the collections HMM_1 ... HMM_5. (a) Sensitivity of set HMM_i in detecting Cas proteins, measured by the number of cassettes found per subtype. Sets HMM_1, HMM_2 and HMM_3 are more fine grained than sets HMM_4 and HMM_5, which detect less Cas proteins overall. (b) Median Accuracy for the classification of subtypes when using set HMM_i with different ML-approaches to determine the evidence for a Cas protein in a cassette. The quality difference is much lower in the overall task of subtype classification compared to the task of detecting individual Cas proteins.

		HMM_1	HMM_2	HMM_3	HMM_4	HMM_5
	No. models	379	385	416	209	201
	No. sequences	14674	14674	23622	16018	16018
(a) Sensitivity per Subtype	I-A	116	116	117	0	0
	I-B	715	715	713	421	421
	I-C	629	629	629	612	612
	I-D	138	138	137	100	100
	I-E	1114	1114	1116	1069	1069
	I-F	354	354	353	339	339
	I-U	136	136	82	8	8
	II-A	320	320	331	249	249
	II-B	28	28	35	35	35
	II-C	327	327	333	328	328
	III-A	376	376	364	326	326
	III-B	292	292	290	178	178
	III-C	93	93	93	83	83
	III-D	184	184	186	49	49
	IV-A	36	36	36	43	43
	V-A	18	18	32	27	27
	VI-A	6	6	4	6	6
	VI-B	40	40	40	40	40
	Total Sensitivy	4922	4922	4891	3915	3915
(b) Accuracy	ERT	0.9900	0.9898	0.9909	0.9907	0.9907
	CART	0.9629	0.9624	0.9636	0.9579	0.9583
	SVM	0.9856	0.9856	0.9830	0.9868	0.9868

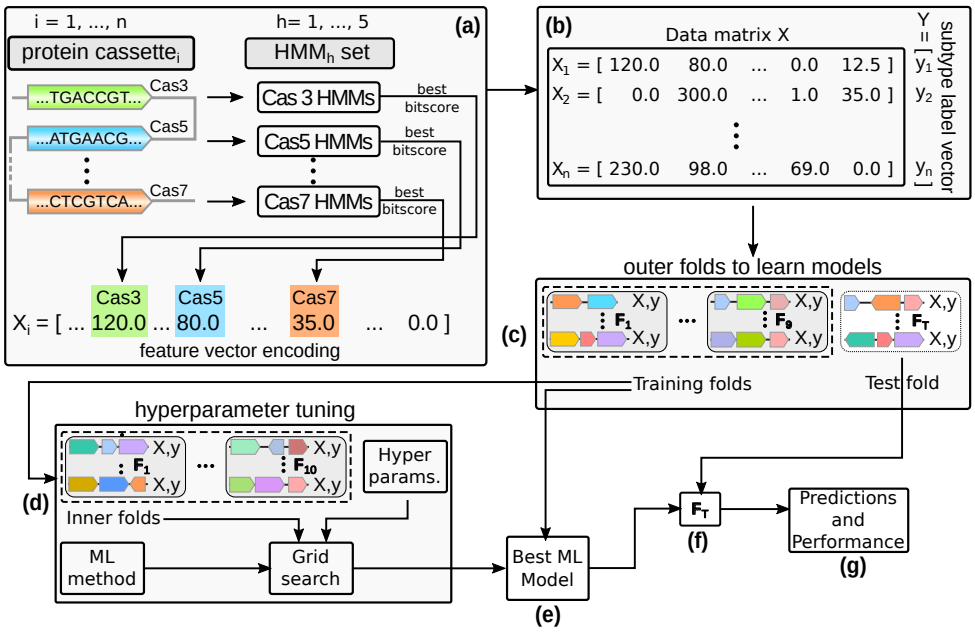
surprising result is that the sensitivity of a specific set HMM_i in detecting Cas proteins does not correlate with the accuracy that is achieved in a subtype classification using this set HMM_i.

3.4.3 A pipeline for CRISPR cassette classification based on Cas protein evidences

Our classification pipeline for CRISPR cassettes is described in Figure 9 and has five steps. For each set HMM_1 ... HMM_5, we build a data matrix for classification and regression analysis of cassettes as follows. Usually, a CRISPR cassette $\mathscr{C}$ is a collection of Cas proteins and is thus defined as a subset of all known Cas proteins $\mathscr{P}$ (i.e., $\mathscr{C} \subset \mathscr{P}$). However, when predicting Cas proteins with HMMer models, this would imply a discretization of the bit score that would omit the information about the *evidence* we have for the prediction. For this reason, we define for each cassette $\mathscr{C}_i$ a real vector X_i of length m, where m is the number of known Cas protein families, containing an entry for each possible Cas protein. Each element X_{ij} is defined as the best bit-score obtained by $\mathscr{P}_j$ among all HMM models of its family if it is detectable by

the models, and zero otherwise (Figure 9a). By concatenating the vectors obtained for all the n available cassettes, we obtain a data matrix $X \in \mathbb{R}_{+}^{n \times m}$ (Figure 9b). In addition, each cassette is associated to a label that indicates its subtype, according to the classification provided by (MAKAROVA *et al.*, 2015; SHMAKOV *et al.*, 2017; SHAH *et al.*, 2018).

Figure 9 – Experimental methodology adopted for this study. (a) Every cassette from our positive set is encoded into a feature vector, which has an entry for each Cas protein family. Given a specific cassette with known Cas proteins, we apply to each Cas protein sequence all HMMs from the set of HMMs that were generated for that specific Cas protein. The best bit-score is included into the feature vector X_i encoding the i^{th} cassette. (b) This feature vector is stored in the data matrix X, together with the known subtype. (c) As the trained model highly depends on the collection of used cassettes, we use the ten-fold cross-validation strategy. Thus, we split the training set into 10 subsets called folds. We perform 10 runs, where, in each run, one of the folds is used for testing and the remaining 9 for selecting and training the best ML model. (d) For selecting the best ML model, a similar cross-validation strategy is applied to tune twenty hyperparameter combinations that affect the model predictive performance. Then, in (e), the selected model is trained using the whole training set. Finally, in (f) and (g) we apply the trained model to the respective test set of the outer fold and evaluate its performance.

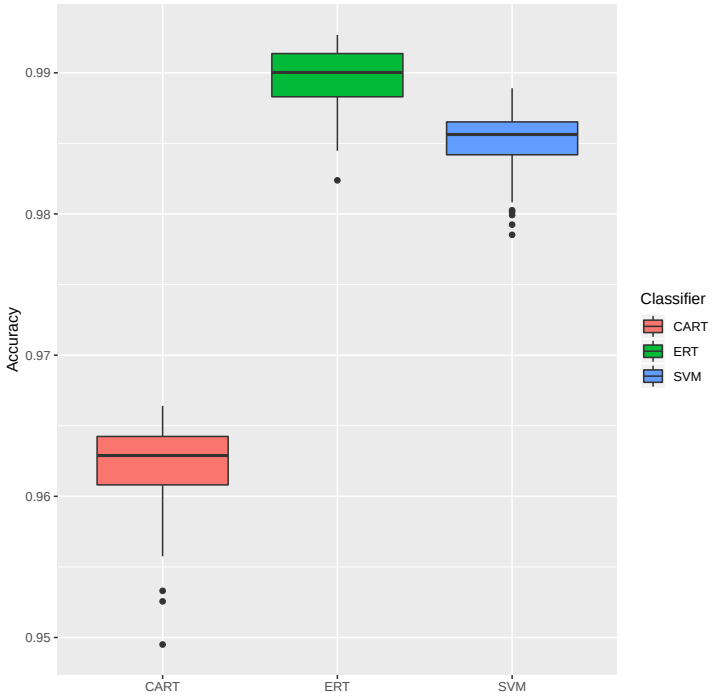


This data matrix, along with the feature vectors and the subtype labels for all known cassettes, is our training data for the subtype classification task. For the evaluation of our classification models, we apply a ten-fold cross-validation procedure on this data matrix. For this, we randomly split the data matrix X into 10 folds (Figure 9c), each one containing a subset of cassettes encoded by the associated feature vector. Each vector is annotated (labeled) by its true subtype. For model selection, we perform hyperparameter tuning by employing a grid search over 20 hyperparameter combinations and applying an inner cross-validation loop (Figure 9d, see Methods for detail). After selecting and training the best model (Figure 9e), we have a classifier that, along with a feature vector with HMM bit scores for all known Cas protein families, predicts the subtype of new cassettes (Figure 9f and Figure 9g).

3.4.4 The classification pipeline successfully predicts the subtype of cassettes

To evaluate the pipeline, we first assessed whether it can successfully perform the classification task, i.e., correctly predict the subtype of a cassette. As shown in Figure 10 for HMM_1, the predictive performances, measured by the adjusted balanced accuracy, for CART, ERT and SVM algorithms are above 95% in general. These high values suggest that, though imbalanced, the cassette subtypes are well-defined in the feature space. It is important to mention that not all cassettes are complete in the investigated datasets. Some cassettes are composed only by subsets of the Cas proteins that integrate its subtype definition. In Supplementary Table 12, we summarize the percentage of cassettes that are complete for each subtype, ignoring Cas proteins that are contained in less than 5% of the cassettes of each subtype. We observed in the experimental results that, even though some incomplete cassettes are present the three classifiers were still able to capture the relations among the remaining proteins. The results for the other four sets of HMM models, and for the F-score with the macro averaging measure were similar and allowed us to draw similar conclusions (see Supplementary Figure 21).

Figure 10 – Adjusted balanced accuracy obtained for the 50 repetitions of nested ten-fold cross-validation applying ML algorithms to the dataset generated by the HMM_1 set. The *x*-axis corresponds to the classifiers trained by different ML algorithms. The *y*-axis shows the range of adjusted balanced accuracy values.



In order to investigate the prediction quality for specific subtypes, we performed an experiment using the *one-vs-the-rest* strategy (BISHOP, 2006). Given k different classes, the *one-vs-the-rest* strategy trains k classifiers, one for each subtype, which learns how to discriminate this subtype (positive class) from the remaining classes (negative class). In Table 6 we show

the average F-scores, after 50 cross-validation repetitions, obtained by the classifiers using the *one-vs-the-rest* strategy. It is clearly visible that the k classifiers were able to discriminate each class with a high predictive performance, in agreement with our previous results. In the case of SVM, one can use the margin separating positive and negative data as an additional quality criterion (CHERKASSKY; DHAR, 2010). Again one can see here a clear separation of SVM scores for the positive and negative classes (see Supplementary Figure 22).

Table 6 – Mean F-scores for 50 nested cross-validation repetitions using the *one-vs-the-rest* strategy and Cas protein set HMM_1.

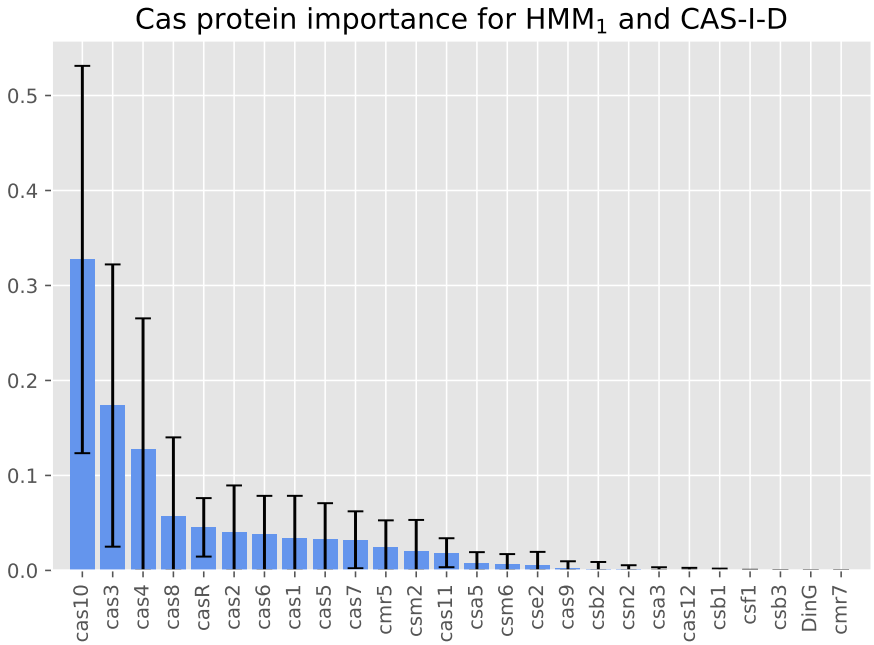
Subtype	CART	SVM	ERT
I-A	0.95	0.96	**0.98**
I-B	0.95	0.98	**0.99**
I-C	0.98	0.99	**1.00**
I-D	0.98	0.97	**0.99**
I-E	0.99	0.99	**1.00**
I-F	0.95	0.99	**1.00**
I-U	0.99	0.97	**1.00**
II-A	**1.00**	**1.00**	**1.00**
II-B	**1.00**	**1.00**	**1.00**
II-C	**1.00**	**1.00**	**1.00**
III-A	0.98	0.98	**0.99**
III-B	0.97	0.98	**0.99**
III-C	0.93	**0.98**	**0.98**
III-D	0.96	0.97	**0.99**
IV-A	**1.00**	**1.00**	**1.00**
V-A	**1.00**	**1.00**	**1.00**
VI-B	0.86	0.95	**0.96**

3.4.5 The classification pipeline detects signature proteins

Makarova *et al.* (2015) define the presence of unique signature Cas proteins that characterize most of the investigated CRISPR subtypes. According to the authors, signatures usually consist of either one or multiple Cas proteins that co-occur in the same cassette. Based on the aforementioned results, we hypothesize that the classifiers were able to learn these signature proteins. Since each *one-vs-the-rest* classifiers introduced in the last section learned how to discriminate a different subtype, we assessed whether it is possible to derive insights about signature proteins for each class by analyzing each classifier separately.

We thus propose a new approach to detect signature proteins for a subtype by determining the importance of a specific feature (i.e, the evidence for a Cas protein in a cassette) to correctly predict the subtype in the respective *one-vs-the-rest* classifier. The rationale is that Cas proteins which are highly important for discriminating a specific subtype against all others are likely signature proteins for this subtype. Figure 11 shows the importance of each Cas protein (see Methods for definition of feature importance) in predicting the I-D subtype. As it can be seen, the importance is specifically high for Cas10(d) (resp. Cas3), which is the signature protein for the subtype I-D (resp. the type I) according to Makarova *et al.* (2015). Overall, we observed that

Figure 11 – *One-vs-the-rest* average Cas protein importance of ERT for I-D subtype. The *x*-axis presents different Cas proteins. The *y*-axis shows the importance of each Cas protein regarding the decision trees of the ensemble split. The error bars refer to the standard deviation over all trees in the ensemble. Note that the feature importance is not only related with the classification into the I-D subtype, but may also be related to its contribution to classify a cassette into any other subtype. Thus, some of the proteins in the figure may not be related to I-D, but to any other subtype.

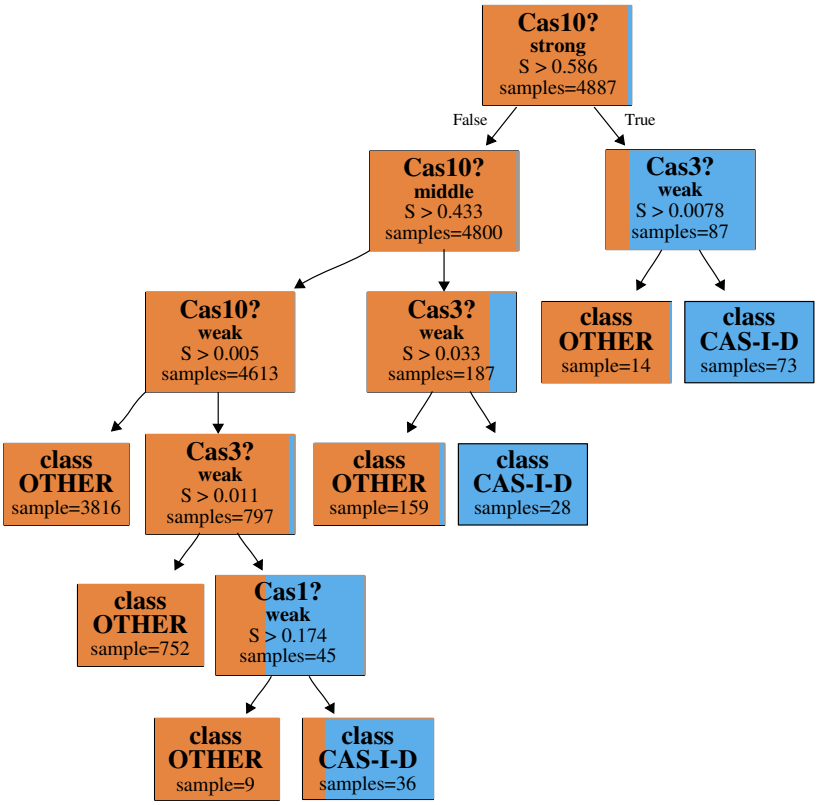


Cas10 and Cas3 account, on average, for more than 50% of the feature importance for classifying the I-D subtype.

To investigate the relation between the two signature genes for proteins Cas10 and Cas3 in more detail, we selected the decision tree obtained by CART for the I-D subtype (Figure 12). In this tree, terminal nodes with the blue color indicate I-D classification (positive class), while those with brown color indicate any other subtype classification (negative class). As shown in Figure 12, Cas10 is the most important protein for identifying I-D, which is in agreement with Makarova *et al.* (2015), where the subtype I-D is characterized by the presence of a variant of the Cas10 protein (instead of a protein from the Cas8 family, which is common for the other I subtypes) and two variants of the Cas3 protein. Interestingly, we need middle to strong evidence for Cas10 and only weak evidence for Cas3. In the case of weak evidence for Cas10, we also need weak evidence for both Cas3 and Cas1 in order to correct the missing 36 examples, albeit in this case the classification would not be pure anymore. Overall, it can be observed that CART was able to correctly model this signature, since most of the nonterminal nodes refer to these proteins, indicating that they are the most important features in this subtype.

Since the current classification (MAKAROVA *et al.*, 2015) is based only on the interference module, the adaptation-related Cas proteins (Cas1, Cas2 and Cas4) should not have

Figure 12 – Reduced *one-vs-the-rest* CART for the I-D class (see Supplementary Figure 23 for full tree). Cassetes that are labeled as subtype I-D are highlighted in blue, the others in brown. Each node shows the fractions of class I-D and other cassetes, indicating the purity of the node. The number of cassetes is shown under the "samples" entry. In each node, we query for evidence of a specific Cas protein, indicated by the score calculated by the HMM family models. As one can see, a strong evidence for Cas10 immediately points to a subtype I-D (top node and right branch). Otherwise, if we have middle evidence for Cas10, we need at least weak evidence for Cas3 to determine subtype I-D. Finally, if we have only weak evidence for Cas10, we need at least weak evidence for Cas3 and also for Cas1 to determine subtype I-D (left branch). However, the classification is not pure anymore (bottom nodes).



a high importance for our classification pipeline. Thus, in another experiment, we removed these proteins and the processing proteins (Cas6), and tested the predictive performance of our classification pipeline when removing this information. The obtained results were similar to those previously discussed in this section and support our discussion and main conclusions (see Supplementary Figure 24), strengthening the hypothesis that our ML-based approach captured biologically relevant information.

All the aforementioned examples illustrate how our ML models are able to learn the protein signatures without any extra information other than the normalized bit scores and cassette subtype labels. These results validate our hypothesis and provide models that are able to automatically categorize new cassettes with a high predictive accuracy.

3.4.6 Regression instead of classification learns association rules

In our next set of experiments, we were interested in answering the question of whether some Cas proteins tend to be co-occurring frequently with other proteins. To answer this question, we hypothesized that they form a functional module. However, as we have varying information about the evidence for a specific Cas protein, and there is also some redundancy and flexibility in forming this module, we followed an approach different from that described in the previous section. We believe that if a specific Cas protein is frequently associated with other Cas proteins, it is possible to predict the evidence for this protein by relying only on the known evidence for the other members of the functional module. We can confirm this belief by removing a specific Cas protein from the feature vector, and predicting the "expected" normalized bit score for this protein from the remaining feature vector. This amounts to learning a regression model from known examples.

Association rules can now be inspected by determining again the important features (i.e., Cas proteins) to predict the correct evidence for a specific Cas protein. In Table 7, we list the three most important proteins for some target Cas proteins in some subtypes. In this case, for predicting evidence for Cas10d in subtype I-D, we need the information about Cas3, Cas5 and Cas7. In agreement with the fact that subtypes are mainly associated with the interference complex (MAKAROVA *et al.*, 2015), we find that for the interference proteins Cas10d, Cas3 and Cse2, the associated proteins are also interference proteins. For the non-interference proteins Csn2 and Cas4 in II-A and II-B, not only is Cas9 an interference and signature protein for type II, but it is associated with them as well as the adaptation proteins Cas1 and Cas2. Interestingly, though Cas9 information is important for Cas4, Cas1 is actually more significant for Csn2. This is in agreement with the hypothesized role of Csn2 in the adaptation process (NAM *et al.*, 2011; KOO *et al.*, 2012; ARSLAN *et al.*, 2013; LEE *et al.*, 2012).

Table 7 – Top 3 most important proteins according to ERT when trying to predict a target protein across different subtypes. For the interference proteins Cas10d, Cas3 and Cse2, the other most important Cas proteins are also interference proteins. For non-interference proteins, other Cas proteins linked to adaptation, e.g. Cas1 and Cas2, are also important. The helper protein CasR seems to have different modules associated in I-A and I-E.

Subtype	Target protein	Most important proteins
I-D	Cas10d	(Cas3, 0.28), (Cas5, 0.26), (Cas7, 0.17)
I-D	Cas3	(Cas11, 0.48), (Cas10, 0.20), (Cas5, 0.11)
I-E	Cse2	(Cas7, 0.25), (Cas5, 0.23), (Cas8, 0.19)
II-A	Csn2	(Cas1, 0.62), (Cas9, 0.23), (Cas2, 0.15)
II-B	Cas4	(Cas9, 0.83), (Cas1, 0.09), (Cas2, 0.08)
I-A	CasR	(Csa5, 0.28), (Cas5, 0.16), (Cas6, 0.12)
I-E	CasR	(Cas1, 0.33), (Cas7, 0.25), (Cas8, 0.21)

An interesting case to consider is CasR (also known as CasRA or Csa3), a transcriptional regulator of CRISPR interference and/or adaptation (HE *et al.*, 2017; VESTERGAARD; GARRETT; SHAH, 2014). This protein seems to have different roles in subtypes I-A and I-E, and also appears to be associated with the different proteins in I-A and I-E (see Table 7, last two

rows). In I-A, the most important proteins are Csa5, Cas5 and Cas6, whereas in I-E they are Cas1, Cas7 and Cas8.

3.4.7 The ML-approach can handle missing Cas proteins

During our experiments, we left out cassettes that had one or more Cas proteins missing, i.e., without hits in their corresponding HMM models during the preprocessing step (Figure 9a). Since these cases often occur in real application scenarios it is important to assess how our ML-based pipeline can handle them. We observed that most of these cassettes contained only one protein that did not present any hit for the HMM models of its family. For such, we worked with the cassettes having all proteins annotated as ground truth, and removed one bit score for a specific protein. We then learned a model able to predict this bit score using the evidence information from the remaining proteins.

Specifically, we investigated the performance of predicting the missing evidence using the previously described regression approach, trained on all subtypes. The basic idea is that a high predicted evidence for a missing protein is a hint for researchers to perform an in-depth attempt to either annotate the missing protein or to search for new proteins that might replace the function of the missing protein.

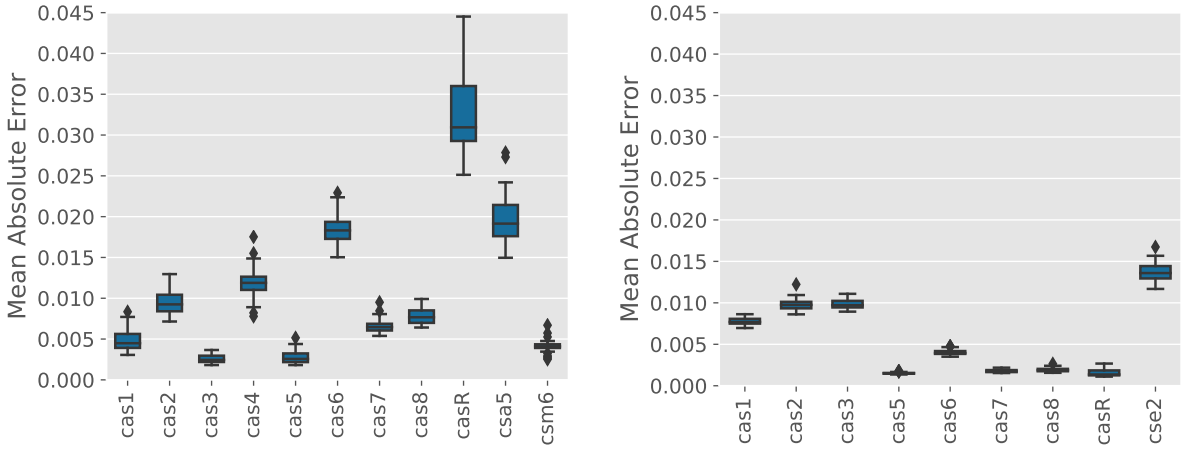
Figure 13 shows the Cas protein regression results for ERT; the regressor with the best predictive performance for subtypes I-A and I-E in the dataset generated by HMM_1. Other experimental results, for different subtypes and datasets, can be seen in our Supplementary material, Figures 25–29. These results show that the missing proteins are predicted with a high quality. For the core proteins Cas1 … Cas10, specifically, the proposed approach has very high prediction rates, showing a strong interdependence between these core proteins and other Cas proteins important for the subtype. We also observed that for proteins that are not core Cas proteins such as CasR, the size of the data basis (i.e., number of known cassettes for the subtype where this protein occurs) influences the prediction quality. While this is partially inherent in the machine learning approach, it also might indicate a more variable or complex interaction between these proteins and other proteins important for the subtype.

We also observed that, in general, ERT obtained the best results for Cas protein regression (see Supplementary Figures 25–29). In most cases, ERT presented mean absolute error values below 0.05 for the normalized bit score prediction. These results confirm the relevance of building specific regressors for each Cas protein inside of a specific subtype for the identification of unknown or possibly missing Cas proteins, when the label of the cassette of interest is known.

In order to assess whether the aforementioned setting would work on a more global level, we replicated the previous experiment by training the regressor on the full datasets with all subtypes. Most of the times, similar results were obtained (see Supplementary Figures 30–34).

Next, as a proof of concept, we looked at the cassettes with one missing protein that

Figure 13 – Mean absolute error rates for Cas proteins contained in I-A (a) and I-E (b) subtypes over 50 nested cross-validation repetitions. The *x*-axis lists the different Cas proteins that were used as target variables. The *y*-axis presents the mean absolute error values between the known bit score, and the bit score predicted by our regression approach. In general, missing proteins are well predicted, especially in the case of the core Cas proteins Cas1 to Cas7. For other Cas proteins, like CasR, the prediction quality varies between I-A and I-E. This is likely due to the higher amount of I-E cassettes in the data basis, indicating a more complex relationship between CasR and other Cas proteins.



(a) I-A (b) I-E

Figure 14 – The cassettes with missing proteins. A) In this genome, we predicted a DinG protein missing in the cassette with evidence > 0.5. The HHblits (REMMERT *et al.*, 2012) search in this genome for all ORFs determined 1 ORF 117nt upstream of the cassette with a high confidence score for a DinG homology (E-value: 7.6e-22). B) In the case of Cas2, the predicted evidence was lower, between 0.221 to 0.165. Nevertheless, we found 1 ORF with a high confidence score for Cas2 homology (E-value: 1e-37) 3010nt downstream of the cassette.

were left out of our experiments and constitute an independent test case (see Methods), and applied our regression approach to identify the cassettes with a high predictive performance of the evidence for having a specific missing protein. These cases would be good candidates for missing annotations. We found 13 cassettes that predicted a missing DinG protein, 3 of them with evidence of at least 0.5. By applying a HHblits (REMMERT *et al.*, 2012) search for all ORFs in the respective genome of these three cassettes, we found an ORF with convincing homology to DinG-proteins in each case (see Figure 14A for an example). Another case was Cas2, when we found 13 cassettes with a missing Cas2 protein predicted. We again used HHblits on all ORFs in the genome of the top three cassettes, and found one case with a convincing Cas2 homology (see Figure 14B).

Finally, we applied our regressors to the aforementioned test set, to predict the missing protein annotations, and the classifiers to predict the subtype for these incomplete cassettes,

Table 8 – Average adjusted balanced accuracy for classification on the independent test set, consisting of cassettes with one Cas protein missing. "Clf." refers to "Classifier" while "Reg." to "Regressor". The best results ≥ 0.7 are in bold. A dash in the second column means no regression (i.e., only classification) was used.

Clf.	Reg.	HMM_1	HMM_2	HMM_3	HMM_4	HMM_5
CART	–	0.50	0.50	0.68	0.48	0.48
	CART	0.68	0.68	0.52	0.55	0.55
	ERT	0.63	0.63	0.56	0.58	0.58
	SVM	0.56	0.56	0.51	0.54	0.54
ERT	–	0.63	0.63	0.65	**0.74**	**0.74**
	CART	**0.70**	**0.70**	0.63	0.64	0.63
	ERT	0.69	0.69	0.63	0.64	0.65
	SVM	0.60	0.60	0.63	0.63	0.62
SVM	–	0.50	0.50	0.58	0.64	0.64
	CART	**0.72**	**0.72**	0.53	0.58	0.58
	ERT	0.66	0.66	0.60	0.61	0.62
	SVM	0.54	0.54	0.53	0.57	0.57

which were not included in the training set. Table 8 shows the classification results for all ML algorithms on this independent test. In these results, the ERT- and SVM-based classifiers, when combined with the CART regressor and the more fine grained models HMM1 and HMM2, can predict the correct subtype with high predictive performance, even in the hard case of incomplete annotation. The ERT-based classifier can also achieve high performance when combined with the less fine grained models HMM4 and HMM5. However, in these cases, there are less subtypes available, because only classes containing at least 10 examples were included in our experiments (see Methods).

3.4.8 CRISPRcasIdentifier clearly outperforms existing tools

Finally, we assessed the quality of prediction in comparison with existing tools, to assess whether they would be able to correctly classify cassettes that are not covered by the manual annotations. This is a typical application scenario e.g. in the analysis of cassettes from metagenomic data. For this purpose, we used CRISPRcasIdentifier with default parameters and compared its performance with three command line CRISPR-Cas tools (CRISPRdisco (CRAWLEY *et al.*, 2018), CRISPRcasFinder (COUVIN *et al.*, 2018) and Macsyfinder (ABBY *et al.*, 2014)) and two webservers (CRISPRone (ZHANG; YE, 2017) and HmmCas (CHAI *et al.*, 2019)).

To benchmark these tools we used the most recent and comprehensive set of cassettes as listed in the very recent classification paper (MAKAROVA *et al.*, 2019). This dataset has 6098 cassettes extracted from 4974 archaeal and bacterial genomes, including the following subtypes: I-A to I-U, II-A to II-C, III-A to III-D, IV-A, V-A and VI-B. In Table 9 we present the adjusted balanced accuracy scores and F-scores with the macro averaging obtained. The inferior results of HmmCas and CRISPRone can partially be explained by the fact that they 1) use the existing Cas HMMer models without any enhancement, and 2) rely on a concept that is similar to signature

Table 9 – Predictive performance of CRISPR-Cas tools for different measures. The best results for each measure are marked in bold.

	Tool	Adj. Bal. Acc.	F-score
web-server	CRISPRone	0.07	0.17
	HmmCas	0.05	0.15
command-line	CRISPRdisco	0.52	0.63
	CRISPRcasFinder	0.48	0.56
	Macsyfinder	0.54	0.60
	CRISPRcasIdentifier	**0.89**	**0.91**

gene for predicting the subtype.

According to Table 9 our tool clearly outperforms the others for all measures. Our hypothesis is that the superior results are due to the generalization capability of ML models. Thus, our tool is more suitable to handle unseen examples even if they contain missing proteins. It occurs because it does not rely only on HMM profile searches but also on the general knowledge extracted from the training data. It is also important to observe that CRISPRcasIdentifier not only classifies unseen cassettes but also tries to predict potentially missing proteins which, to the best of our knowledge, is a problem that has not been successfully addressed by the existing tools.

3.5 Conclusions

In this paper we introduced a new ML-based pipeline for the identification and classification of genomic CRISPR-Cas systems. To assess the predictive performance of this approach, we conducted an in-depth investigation into the suitability of ML algorithms that are commonly used for this task, by using the normalized profile HMM search bit scores of Cas proteins as input and classifying cassettes encoding Cas proteins to their respective subtypes according to the most recent classification (MAKAROVA *et al.*, 2015; SHMAKOV *et al.*, 2017; SHAH *et al.*, 2018).

Overall, this work covers four different research issues: (i) the classification of Cas cassettes; (ii) the prediction of normalized bit scores for missing Cas proteins; (iii) the investigation of the properties of CRISPR types and subtypes; and (iv) the comparison of our new tool to the ones available in the literature. Concerning topic (i), our classification models were able to achieve very high classification performance, above 0.95, in terms of the adjusted balanced accuracy score. Thus, they are well placed for the prediction of CRISPR systems of newly sequenced organisms, or metagenomic data with sufficient read length to cover the full cassette in one contig. In addition, we introduced a new method for determining signature genes, which are genes most important for predicting the correct subtype. This approach was able to properly learn the known signature genes of CRISPR-Cas subtypes without any extra information other than the available gene cassettes and their labels, but provides additional information about the composition of

cassettes. In topic (ii), our regressor models achieved very small deviations between the expected and predicted normalized bit scores for different Cas proteins across the different subtypes. This illustrates the usefulness of these regressors on new cassettes that have missing hits for some Cas proteins. A high bit score provides a hint to researchers to search for more diverged forms of the protein, or to look for proteins which could replace the missing function. The analysis performed under topic (iii) enabled us to correctly identify known signature genes, and to identify putative functional modules. Overall, it provided us with a set of association rules for potential use in more advanced classification scenarios, in addition to providing insights about the biology of the systems. Finally, concerning (iv), our tool outperformed five other tools from the literature on the most recent and comprehensive CRISPR classification dataset published.

Manual annotation is the gold standard when it comes to classification and identification of genomic CRISPR-Cas systems. Supporting this process or annotating cassettes as part of an overall automatic pipeline such as the analysis of metagenomic data requires a classification approach with a degree of flexibility that is challenging to model. CRISPRcasIdentifier provides a boost in classification accuracy when compared to existing tools, because it builds not only on an understanding of the manual annotation process but also on the generalization power of ML algorithms. We made CRISPRcasIdentifier available for researchers to use with their own data.

3.6 Availability of Source Code and Requirements

Project name: CRISPRcasIdentifier
Project home page: <https://github.com/BackofenLab/CRISPRcasIdentifier>
RRID: SCR_018296
BiotoolsID: crisprcasidentifier
Operating system(s): Platform independent
Programming language: Python
Other requirements: Anaconda, Docker
License: GNU General Public License version 3 (GPLv3)

3.7 Availability of Supporting Data and Materials

The data that supports the present work are available from the studies of (MAKAROVA *et al.*, 2015; SHMAKOV *et al.*, 2015; SHMAKOV *et al.*, 2017; MAKAROVA *et al.*, 2019). An archival copy of the code and supporting data are also available via the GigaScience database GigaDB (PADILHA *et al.*, 2020b).

CHAPTER

4

CASBOUNDARY: AUTOMATED DEFINITION OF INTEGRAL CAS CASSETTES

Publication information: This chapter is an article that was published in the Bioinformatics journal by Oxford University Press. The article is licensed under CC BY 4.0*.

Reference: PADILHA, V. A.†; ALKHNBASHI, O. S.†; TRAN, V. D.; SHAH, S. A.; DE CARVALHO, A. C. P. L. F.; BACKOFEN, R. Casboundary: Automated definition of integral Cas cassettes. **Bioinformatics**, 2020. ISSN 1367-4803. Btaa984. Available: <https://doi.org/10.1093/bioinformatics/btaa984>.

4.1 Abstract

CRISPR-Cas are important systems found in most archaeal and many bacterial genomes, providing adaptive immunity against mobile genetic elements in prokaryotes. The CRISPR-Cas systems are encoded by a set of consecutive *cas* genes, here termed cassette. The identification of cassette boundaries is key for finding cassettes in CRISPR research field. This is often carried out by using Hidden Markov Models and manual annotation. In this paper, we propose the first method able to automatically define the cassette boundaries. In addition, we present a Cas type predictive model used by the method to assign each gene located in the region defined by a cassette's boundaries a Cas label from a set of pre-defined Cas types. Furthermore, the proposed method can detect potentially new *cas* genes and decompose a cassette into its modules. We evaluate the predictive performance of our proposed method on data collected from the two most recent CRISPR classification studies. In our experiments, we obtain an average similarity of 0.86

*<https://creativecommons.org/licenses/by/4.0/>

† Contributed equally as first authors.

between the predicted and expected cassettes. Besides, we achieve F-scores above 0.9 for the classification of cas genes of known types and 0.73 for the unknown ones. Finally, we conduct two additional study cases, where we investigate the occurrence of potentially new *cas* genes and the occurrence of module exchange between different genomes.

4.2 Introduction

Prokaryotes face tremendous evolutionary pressures from viral predators, such as bacteriophages, which are responsible for eradicating almost half of the earth's bacterial population each day (SUTTLE, 2016). This constant threat has been hypothesised to comprise the single most important driver of the planet life evolution (KOONIN *et al.*, 2020). Bacteria and archaea face an enormous incentive to defend themselves against viral invaders by evolving defense systems, some of which are innate and others adaptive. Clustered Regularly Interspaced Short Palindromic Repeats (CRISPRs) constitute one such nucleic acid based adaptive immune system, which functions through three distinct stages: acquisition, processing and interference. Upon a naive infection, a piece of viral nucleic acid is incorporated as a spacer between the repeats of the CRISPR locus on the host chromosome during its acquisition. The whole CRISPR locus, which includes memories from dozens of past viral infections, is transcribed into a long piece of RNA that is processed into small mature CRISPR RNAs (crRNAs), each corresponding to a different acquired viral epitope. crRNAs are loaded onto the Cas (Crispr ASsociated) interference complex, which then scans all intracellular nucleic acid for a matching nucleotide sequence, in which case the target nucleic acid is cleaved, effectively protecting the cell from reinfection by any virus for which a matching spacer exists.

Bacteriophages and archaeal viruses evade CRISPR immunity by several mechanisms. Known mechanisms include direct mutations of the nucleic acid such that it is no longer targeted by the host (HORVATH *et al.*, 2008), or the evolution of small anti-CRISPR proteins. These proteins interfere with the proper function of the Cas proteins that mediate CRISPR immunity by either clogging catalytic sites or preventing complex assembly. Hosts evade such anti-CRISPR immunity by carrying several distantly related CRISPR-Cas systems at once, and by frequently exchanging their CRISPR-Cas systems for different ones through horizontal gene transfer. This dynamic has driven the diversification of CRISPR-Cas systems into six types that are further subdivided into 33 subtypes (MAKAROVA *et al.*, 2019), each with its own evolutionary trajectory. Corresponding Cas protein subunits from two different hosts, even when belonging to the same subtype, can have sequences so distant that they are unalignable despite sharing the same underlying protein structure. Such extreme diversification is caused by Cas proteins mutating in order to avoid being inactivated by phage anti-CRISPRs. The rapid evolution of CRISPR-Cas systems makes their detection difficulty in metagenomic sequences of uncultured bacteria and archaea, because none of the existing known CRISPR-Cas systems in completely sequenced genomes is a close enough match. Although the new Cas proteins are structurally

similar to known Cas proteins, the amino acid sequences have diverged to an extent that makes them difficult to detect even using the most sensitive sequence alignment methods (REMMERT *et al.*, 2012). While some Cas proteins such as Cas1 are easy to detect due to its very conserved sequence, other proteins, such as Cas8, are notoriously difficult to identify, owing to their strong sequence heterogeneity. Thus, even the most modern bioinformatics pipelines for annotation of genomic CRISPRCas loci have difficulties in detecting all cas genes comprising a complete CRISPR cassette.

According to comparative genomics studies of chromosomally encoded CRISPR-Cas systems (GARRETT *et al.*, 2011; VESTERGAARD; GARRETT; SHAH, 2014; MAKAROVA *et al.*, 2015; SHAH *et al.*, 2018), these systems are carried on genomic cassettes, which are further divided into modules corresponding to the different functional stages of the immune response. Cassettes, as well as modules, are normally integral, meaning they have defined boundaries and are not intermixed with foreign genes. Thus, a typical bacterium may carry several Cas cassettes, and each cassette can be further divided into several operons, each corresponding to a functional module. Class 1 systems, in particular, have elaborated heteromultimeric interference complexes typically consisting of between four and eight genes. Knowing where the module starts and ends on the genome narrows down the possibilities and is an invaluable aid in annotating the *cas* genes that do not yield matches to any known Cas proteins.

Current bioinformatics pipelines for annotating *cas* genes treat each gene separately, while a cassette-aware pipeline could infer the identities of missing genes by simple exclusion (LANGE *et al.*, 2013; ALKHNBASHI *et al.*, 2014; ALKHNBASHI *et al.*, 2016; ZHANG; YE, 2017; CRAWLEY *et al.*, 2018; COUVIN *et al.*, 2018; ALKHNBASHI *et al.*, 2020).

In this paper, we propose a new method to automatically define the boundaries of a CRISPR cassette. The proposed method takes into account the relation of a potential signature gene and genes that are contained in its neighboring region. Furthermore, the method labels the *cas* genes after the cassette boundaries have been defined, being also able to indicate genes that may belong to new putative types.

4.3 Methods

This section introduces notation, definitions and problems addressed in this paper. Afterwards, it describes our proposed method for cassette boundary detection and Cas type prediction in details.

4.3.1 Problem statement and notations

For a given genome, let $g_1, \ldots, g_n$ be all genes of the genome ordered by its genomic location (i.e. g_i is located between g_{i-1} and g_{i+1} on the genome), and let $\mathcal{G} = \{g_i | i \in [1:n]\}$ be

the set of all genes in the genome. With $\mathscr{G}_c$ we denote the set of all *cas* genes in this genome, and the set of all *cas* signature genes by $\mathscr{G}_s$. $\mathscr{G}_u = \mathscr{G} \backslash \mathscr{G}_c$ is the set of all *non-cas* genes.

We denote $S_{ij} \subseteq \mathscr{G}$ as a set of consecutive genes $S_{ij} = \{g_i, \ldots, g_j\}$ and the set of all its consecutive subsets as $\mathbf{Sub}(S_{ij})$. Note that $\mathbf{Sub}(S_{ij})$ is not exponential in size as we are considering only subsets that contain all genes in a genomic region. A consecutive subset C is called a *cassette* if it contains a sufficient number of *cas* genes and not too long stretches of *non-cas* genes. Formally, $C = S_{pq}$ is a cassette if

1. $g_p \in \mathscr{G}_c$ and $g_q \in \mathscr{G}_c$ (first and last gene is a *cas* gene)
2. $g_{p-1} \notin \mathscr{G}_c$ and $g_{q+1} \notin \mathscr{G}_c$ (the cassette is maximal)
3. $p - q + 1 \geq 3$ (the cassette contains at least three genes)
4. $\forall U \in \mathbf{Sub}(S_{pq}) : U \subseteq \mathscr{G}_u \rightarrow |U| \leq 3$ (each consecutive subset of non-cas genes (called *gap*) is smaller than 3).

We call g_p and g_q lower bound and upper bound of the cassette, respectively. A cassette is often recognized by the presence of its signature gene, g_i^s. The set of all cassettes is notated as $\mathscr{G}_{cs}$.

We formalize the problems addressed in this paper as follows:

- *Cassette boundary detection*: in the first task, we aim at detecting the boundary for each cassette, given its signature *cas* gene. For such, we define a function $f_c(R, g_i^s)$ that takes a potential region R and a signature gene $g_i^s \in R$ as its input and returns the boundaries of the maximal cassette $S_{pq} \in \mathscr{G}_{cs}$ with $S_{pq} \subseteq R$ and $g_i^s \in S_{pq}$.
- *Cas type prediction*: in the second task, we want to determine the label for every *cas* gene. Formally, we define function $f_l : \mathscr{G}_c \rightarrow L \cup \{N\}$, which maps a *cas* gene in $\mathscr{G}_c$ to a label in $L \cup \{N\}$, where L is the set of known Cas labels (such as Cas1, Cas2 etc.) and N is the label for unannotated *cas* genes.

4.3.2 Detection of cassette boundaries

In this section we describe our proposed method for cassette boundary detection implementing the function f_c. Our method is based on our assumption that the relation between a *cas* gene in a cassette and its signature gene is stronger than the relation for a gene that does not belong to that cassette. This assumption is motivated by the common understanding that signature genes $g^s \in \mathscr{G}_s$ play an important role in defining the cassettes (MAKAROVA *et al.*, 2015; MAKAROVA *et al.*, 2019) and should be used as an anchor point in learning the cassette detection function f_c. Furthermore, to simplify the problem of cassette detection, we define an auxiliary function $f(g_j, g_i^s)$ that is 1 (positive) if both genes are located in the same cassette and

0 (negative) otherwise. Thus, the first step of our method is to train a binary classification model for this auxiliary function f. We then use this trained model to detect cassette boundaries in an incremental manner as follows.

First, we slide over the genome. Whenever a signature gene g_i^s is identified, a potential region, R, is defined for detecting the cassette boundary as $R = S_{i-k,i+k}$, where $k > 0$ is large enough such that the full cassette is located inside this region. Next, the model is applied to predict the label for every tuple $(g_j, g_i^s),\ g_j \in R$, starting from the genes located next to g_i^s and extending the range in a stepwise manner. Finally, the boundaries p, q are predicted by Algorithm 1.

Theorem. Let $R = S_{i-k,i+k}$ be the region around a signature gene g_i^s and S_{pq} be the associated cassette predicted by Algorithm 1. Then $S_{pq} = f_c(R, g_i^s)$.

Proof. Let $f_c(R, g_i^s) = S_{s,t} \in \mathscr{G}_{cs}$. First we note that $R \cap S_{pq} \neq \emptyset$ as both R and S_{pq} contain g_i^s. To show equality, we proof by contradiction that there are no left-handed differences. The right-handed cases are analogous. Now lets assume that $s < p$. In this case, let r be maximal in $s \leq r < p \leq i$ such that $g_r \in \mathscr{G}_c$, which must exist as $g_s \in \mathscr{G}_c$ by definition of a cassette. Then $U = \{g_{r+1}, \ldots, g_{p-1}\} \subseteq \mathscr{G}_u$ by construction. As $S_{s,t}$ is a cassette and $g_r \in S_{s,t} \wedge g_i^s \in S_{s,t}$, we know that $f(g_r, g_i^s) = 1$ and $|U| \leq 3$. Hence, g_r would have been detected on the first loop of Algorithm 1 as it started from position $i > p$ and must have considered position p, which is a contradiction.

For the other case let's assume $p < s$. Note that s must have been visited in the first loop of Algorithm 1 as $s \leq i$. Let be g_r be a *cas* gene with $p \leq r < s \leq i$ and r maximal. This must exist as $g_p \in \mathscr{G}_c$ by the stop condition of the first loop in Algorithm 1. Let $U = \{g_{r+1}, \ldots, g_{s-1}\}$. By the first loop of algorithm 1, we know $f(g_r, g_i^s) = 1$ and $|U| \leq 3$. Thus, $S_{r,t} \supset S_{s,t}$ is a larger cassette, which is a contradiction to the maximality of $S_{s,t}$.

Finally, we get $s = p$ and analogously $t = q$, which proofs our claim.

4.3.3 Classification of Cas proteins

Given the boundaries of a cassette, it is important to know the type of each *cas* gene in the cassette. A *cas* gene may belong to a set of predefined types or to a new type (i.e., previously undefined). To create a model able to identify the type of a *cas* gene, we train a multiclass classification model whose output indicates the probabilities of a given *cas* gene to belong to each Cas type. For such, we follow the procedure for word classification proposed in Shu, Xu and Liu (2017), briefly described next:

1. We assume that the probability values of all examples g_i that belong to each class C_j are normally distributed and centered at $\mu(C_j) = 1$. To create the other half of the distribution, we mirror each of these probability values around $\mu(C_j)$ (i.e., for each prob-

Algorithm 1 – Detection of CRISPR boundaries.

Input:
- f: Auxiliary model,
- k: Potential region size parameter,
- $R = S_{i-k,i+k}$: The potential region,
- g_i^s: Signature gene.

Output: $\mathscr{C} \subseteq R$: Cassette

begin
  init: $r = 1$, $p = 0$, gap $= 0$;
  while $r \leq k$ *and* gap ≤ 3 **do**
    if $f_c(g_{i-r}, g_i^s) = 1$ **then**
      $p = r$;
      gap $= 0$;
    else
      gap $=$ gap $+ 1$;
    end if
    $r = r + 1$;
  end while
  init: $r = 1$, $q = 0$, gap $= 0$;
  while $r \leq k$ *and* gap ≤ 3 **do**
    if $f_c(g_{i+r}, g_i^s) = 1$ **then**
      $q = r$;
      gap $= 0$;
    else
      gap $=$ gap $+ 1$;
    end if
    $r = r + 1$;
  end while
  $\mathscr{C} = S_{i-p,i+q}$;
  if $|\mathscr{C}| < 3$ **then**
    return $\emptyset$;
  end if
  return $\mathscr{C}$;
end

ability value $P(C_j|g_i)$ associated to a training example g_i, we create the artificial point $1 + (1 - P(C_j|g_i)))$.

2. We estimate the standard deviation $\sigma(C_j)$ using the obtained probabilities and the artificial mirrored values.

3. Finally, for each class C_j, if the predicted probability for a test example g_k is below the threshold $t(C_j) = \max(0.5, 1 - \alpha\,\sigma(C_j))$, g_k is considered as an outlier for C_j. If the example is considered as an outlier for all classes, we label it as N (unnanotated). As suggested by Shu, Xu and Liu (2017), we used $\alpha = 3$. Otherwise, if the *cas* gene is not considered as an outlier, we assign to it the label with the highest probability.

In the original paper, Shu, Xu and Liu (2017) used the training examples to estimate all thresholds $t(C_j)$. However, in our study, we found out that this approach may yield overly optimistic estimations. To overcome this limitation, we used instead a validation set to estimate the thresholds.

4.3.4 Cassette modularization

Earlier studies (GARRETT *et al.*, 2011; SHAH *et al.*, 2011; VESTERGAARD; GARRETT; SHAH, 2014) have found that Cas cassettes can be subdivided into discrete functional modules, with each module carrying out a separate function, and with its genes being spatially separate from other modules within the cassette. Annotating the constituent modules inside a cassette can reveal important information in terms of the functional organisation of the CRISPR-Cas system. Typically, a cassette is composed by three types of modules: adaptation, processing and interference. The processing module typically consists of a single *cas* gene, which is located either close by the interference module or far away from the region defined by the cassette boundaries. For these reasons, we take only the adaptation and interference modules into account. The adaptation module contains genes that are the most conserved across different genomes, being easy to detect. Therefore, in the first step of our method, we want to detect the adaptation module by searching for a sub-region containing Cas1, Cas2 and/or Cas4. Next, the sub-regions which are adjacent to the adaptation module will be considered as the interference modules.

In CRISPR-Cas field, a cassette can have one or more interference modules. Based on the number of interference modules, we define cassettes with a single interference module as *single cassettes* and cassettes with more than one interference module as *multi-module cassettes*. Note that the interference modules in a multi-module cassette might be overlapped or separated.

4.4 Empirical evaluation

4.4.1 Data collection and preprocessing

We collect CRISPR data publicly available from Makarova *et al.* (2015), Makarova *et al.* (2019). Our dataset has 52730 Cas proteins, with 7793 CRISPR cassettes distributed into 22 different subtypes (see Supplementary Table 30). We download the genomes from the NCBI database and extract the Cas protein sequences by applying the Prodigal tool v2.6.3 (HYATT *et al.*, 2010) on the respective gene sequences. For each CRISPR cassette, we identify its signature gene g_i^s, the most important gene to define the cassette of interest (MAKAROVA *et al.*, 2015; MAKAROVA *et al.*, 2019). Next, we extract k genes downstream and k genes upstream to g_s. Usually, the length of a CRISPR cassette ranges from 3 to 15 genes (MAKAROVA *et al.*, 2015; MAKAROVA *et al.*, 2019). Thus, we set $k = 50$, which safely includes the full cassette in the extracted region.

To define the features for each gene, we use three different types of features, described as follows:

1. *General HMM features*: we collect all available Hidden Markov Models (HMM) from the following public databases: TIGRFAM (HAFT; SELENGUT; WHITE, 2003), Pfam (BATEMAN *et al.*, 2004), COG (TATUSOV *et al.*, 2000) and CDD (MARCHLER-BAUER *et al.*, 2010), totalizing 38847 HMMs. For each protein sequence, the features are defined as the bitscores generated by each HMM. We reduce the number of features to 500 using the Truncated Singular Value Decomposition (MANNING; RAGHAVAN; SCHÜTZE, 2009), with 60% of the original data variance preserved.

2. *Protein properties features*: we calculate 12 features related to the properties of each extracted protein, such as: molecular weight, length, isoelectric point, number of negatively charged residues, number of positively charged residues, extinction coefficient (with and without cysteine), instability index, hydrophobicity and secondary structure properties (fraction of turn, sheet and helix).

3. *Specific HMM features*: we build 623 HMM models for the different Cas protein models based on the core and signature genes from the dataset used (MAKAROVA *et al.*, 2015; MAKAROVA *et al.*, 2019). Since these HMM models are more specific to the CRISPR domain, we believe that they may be better suited for the task of identifying potentially new Cas types.

We create a dataset of 7793 cassettes, out of which 7687 are single cassettes, such as those illustrated in Figure 15. Each one of the remaining 106 cassettes, which are multi-module cassettes, can be decomposed into two or three single cassettes whose signatures are close in the genome. We divide these 106 cassettes into 2 subgroups: (i) the *Separated set*, which contains 74 multi-module cassettes that can be broken up into 145 single cassettes that do not overlap (e.g., see Figure 16a); and (ii) the *Overlapped set*, which contains 32 multi-module cassettes that can be broken up into 70 single cassettes that present some degree of overlap (e.g., see Figure 16b).

Figure 15 – Examples of the structure of CRISPR cassettes: (a) single CRISPR cassette; and (b) single CRISPR cassette with a gap. The signature genes are in bold. Blue arrows are interference genes while purple arrows are adaptation genes.

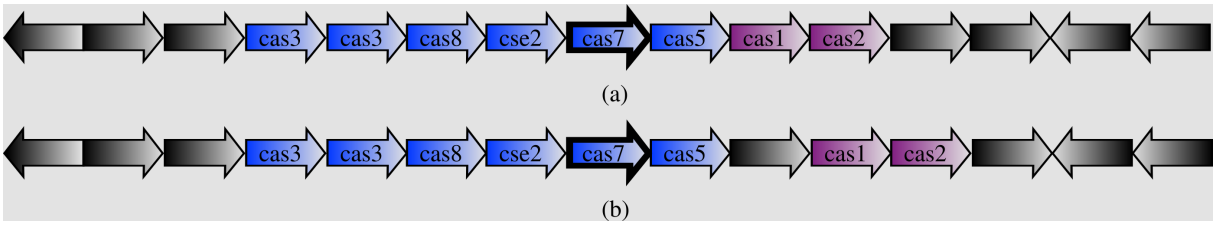
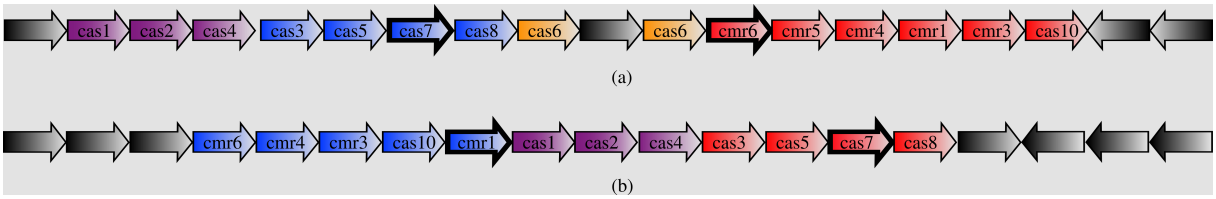


Figure 16 – Examples of the structure of multi-module CRISPR cassettes: (a) multi-module cassette without overlap; and (b) multi-module cassette with overlap. The signature genes are in bold. The blue and red arrows are interference genes, yellow arrows are processing genes and purple arrows are adaptation genes.



4.4.2 Machine learning algorithms

Our method for cassette boundary detection requires a binary classification model, whereas the Cas type prediction demands a multiclass classification model. In our experiments, we use two algorithms to train them which, in addition to be known for their good performance in several tasks, have different learning biases:

- *Extremely Randomized Trees (ERT)* (GEURTS *et al.*, 2006), which is a classifier that integrates multiple decision trees in an ensemble. To define the splits for each tree, this method selects, at each step, a random subset of v features and a subset of v thresholds (one for each feature). Afterwards, the feature that contains the best randomly chosen threshold according to the quality criterion is selected. After training, the class predicted for unseen examples is defined by the majority vote of all trees.

- *Deep Neural Networks (DNNs)* (GOODFELLOW; BENGIO; COURVILLE, 2016), which are neural networks with a large number of layers whose neurons' total input is a dot product between a numeric vector input and the neuron's synaptic weights followed by the application of a non-linear activation function. By using the first layers to extract relevant features, DNNs can learn highly complex functions. DNNs are usually computationally expensive to train. However, with the recent advances in the computer processing power, they have obtained the best predictive performance in a wide range of applications (LIU *et al.*, 2017).

4.4.3 Experimental setup

Two experiments are carried out to evaluate the predictive performance of the proposed method. The first assesses the ability of our method to detect cassette boundaries. For such, we use 10-fold cross-validation for the dataset with 7687 single cassettes, separating one of the training folds for validation, and hold-out for the dataset with 106 multi-module cassettes. The second experiment evaluates how well the proposed method classifies Cas proteins. In this experiment, we employ hold-out for a dataset with 52730 Cas proteins.

4.4.3.1 Cross-validation.

We split the data into 10 folds. Before training, we undersample the majority (negative) class, to mitigate the negative effect of data imbalance on the model training. We repeat the experiment 10 times and report the average and standard deviation of the performance over the 10×10 runs.

4.4.3.2 Hold-out.

For the Cas type classification, we leave 20% of the data out for testing and the remaining for training (80%) and a fifth of the training set, for validation. To evaluate the performance for undefined *cas* genes, we leave in turn 1 and 3 Cas types out of the training and validation set to simulate undefined Cas types scenarios. We repeat this procedure to ensure that every Cas type is left out once. We run the experiment 10 times. Regarding the boundaries detection for multi-module cassettes, we use the 7687 single cassettes as a training and validation set and the 106 multi-module cassettes as the test set.

4.4.3.3 Model selection.

To tune the hyperparameters of each learning algorithm, we employ the grid search with 32 different hyperparameter combinations. For ERT, we tune the number of trees in $\{50, 100, 150, 200\}$, the number of features randomly selected for each split in $\{\sqrt{m}, \log_2 m\}$ and the minimum number of examples to be at a leaf node in $\{1, 4, 7, 10\}$. For DNNs, we use two hidden layers and vary their numbers of neurons in $\{25, 50, 75, 100\}$, the Adam optimizer (KINGMA; BA, 2015) and consider the learning rate values in $\{0.01, 0.001\}$. Concerning maximum gaps, we consider values between 0 and 3.

4.4.3.4 Evaluation metrics.

For the evaluation of cassette boundaries detection, we use the following measures:

- The Jaccard Similarity (JS), which is a popular measure for comparing different sets and is defined as:
$$\mathrm{JS}(\mathscr{C}^t, \mathscr{C}^p) = \frac{|\mathscr{C}^t \cap \mathscr{C}^p|}{|\mathscr{C}^t \cup \mathscr{C}^p|},$$
where $\mathscr{C}^t$ and $\mathscr{C}^p$ are the true and the predicted cassette, respectively. This measure lies in the interval $[0, 1]$ where 1 indicates a perfect match.

- The Cassette Loss (CL), which is an adaptation of the mean absolute error, a popular measure for the evaluation of regression tasks. CL quantifies the gene-wise mean absolute error and is defined as:
$$\mathrm{CL}(\mathscr{C}^t, \mathscr{C}^p) = \frac{|p^t - p^p| + |q^t - q^p|}{2},$$

where p^t (resp. p^p) and q^t (resp. q^p) refer to the index of the first and the last gene of the true (resp. predicted) cassette, respectively. This measure lies in the interval $[0, \infty)$ where 0 indicates a perfect match, i.e., the boundaries of the predicted cassette are in perfect agreement with true cassette. Intuitively, CL denotes the average boundary deviation for the left and right end together.

For the evaluation of the Cas protein classification, we use the F-score with macro-averaging. Given a binary classification task where we have a specific class of interest (positive class), the classical F-score is defined as:

$$\text{F-score} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}$$

where TP, FP and FN correspond to the number of true positives, false positives and false negatives, respectively. For the multiclass scenario, the macro-averaging consists of calculating the F-score for each individual class and reporting the average F-score as the global performance measure. The main advantage of macro-averaging is that it treats all classes with the same weight, independently of the number of examples that they contain (SOKOLOVA; LAPALME, 2009).

4.5 Results and discussion

In this section we report and analyze the results obtained from our experiments.

4.5.1 Detection of cassette boundaries

We report the histogram of JS and CL values for single cassette prediction in Figure 17, using only the general HMM features, which were our best results. For the histograms of other types of features, please check our Supplementary Material (Figures 35 and 36). From Figures 17a and 17c, it can be noticed that most of the JS values are 1.0 and CL values are 0.0, indicating that our model is able to correctly predict most of the cassettes. In addition, in Table 10, we show the average JS and CL values that we obtained for both single and multi-module cassettes. When comparing our results to those achieved by CRISPRCasFinder (COUVIN *et al.*, 2018), the closest tool to our method, it is possible to note that we achieved around 16% of JS improvement in the best case for single cassettes. In particular, our tool would predict cassette boundaries correctly with a precision of roughly one position, whereas CRISPRCasFinder would be roughly 5 positions away on average. Regarding multi-module cassettes, we obtained JS values above 0.70, while CRISPRCasFinder achieved extremely low JS values which are less than 0.15 in both separated and overlapped cases. It confirms the superiority of our method over CRISPRCasFinder in the detection of cassette boundaries. Besides, to illustrate the capability of our method in this scenario, we present in Figure 18 an example of cassette prediction for the organism Thermotoga sp. RQ2.

Figure 17 – Histogram containing 100 equally sized bins of the Jaccard Similarity and Loss for single cassette prediction using ERT (a, b) and DNN (c, d). The inner figures are the zoom of the corresponding outer ones without considering the most dominant bin.

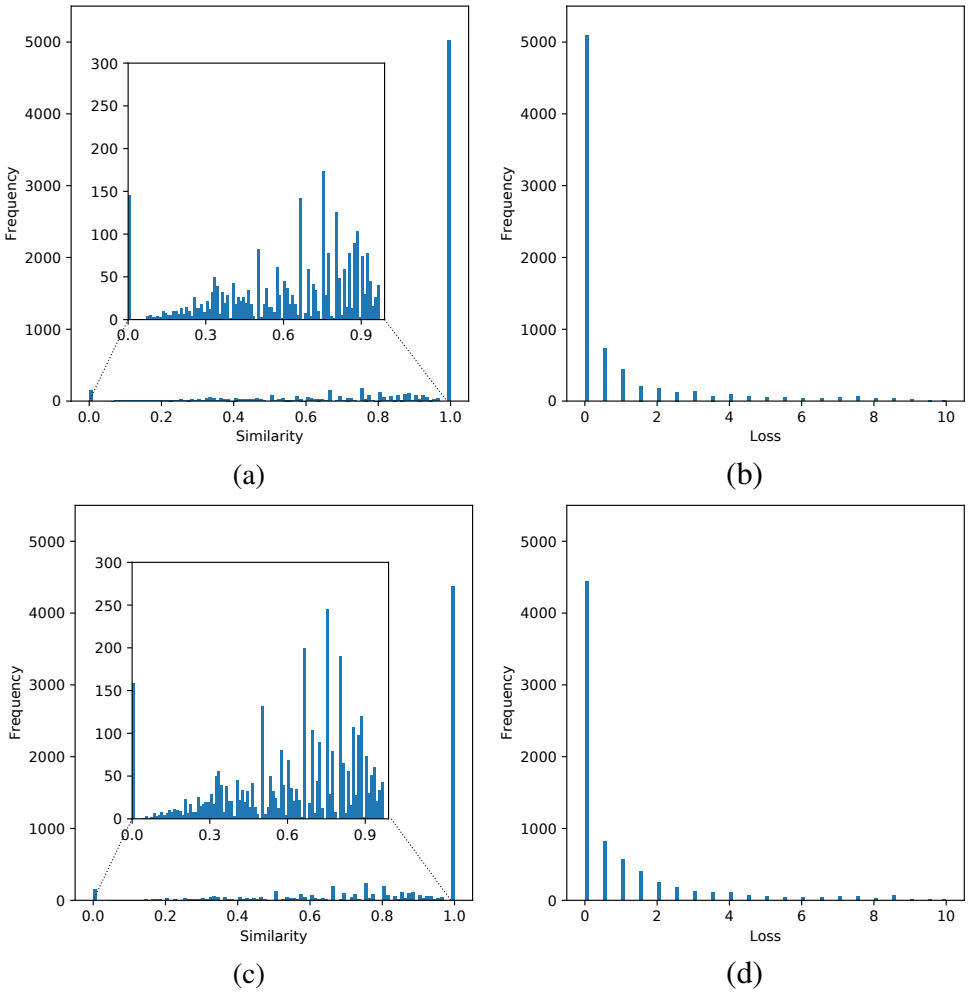


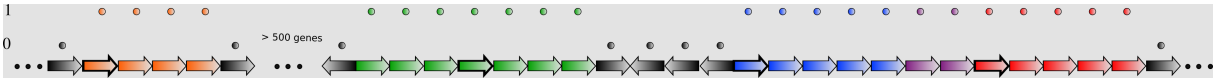
Table 10 – Performance of our method and CRISPRcasFinder for the identification of single and multi-module cassettes in terms of JS and CL. For multi-module cassettes, the prediction quality for boundary detection drastically drops for CRISPRcasFinder, whereas our tool has similar performance to the single cassette case.

Method	Single cassettes		Multi-module cassettes			
			Separated set		Overlapped set	
	JS	CL	JS	CL	JS	CL
ERT	0.86 ± 0.01	1.09 ± 0.12	0.79	1.10	0.72	1.93
DNN	0.83 ± 0.01	1.39 ± 0.20	0.74	1.77	0.73	2.21
CRISPRcasFinder	0.70	4.87	0.13	30.52	0.10	19.88

4.5.2 Classification of Cas proteins

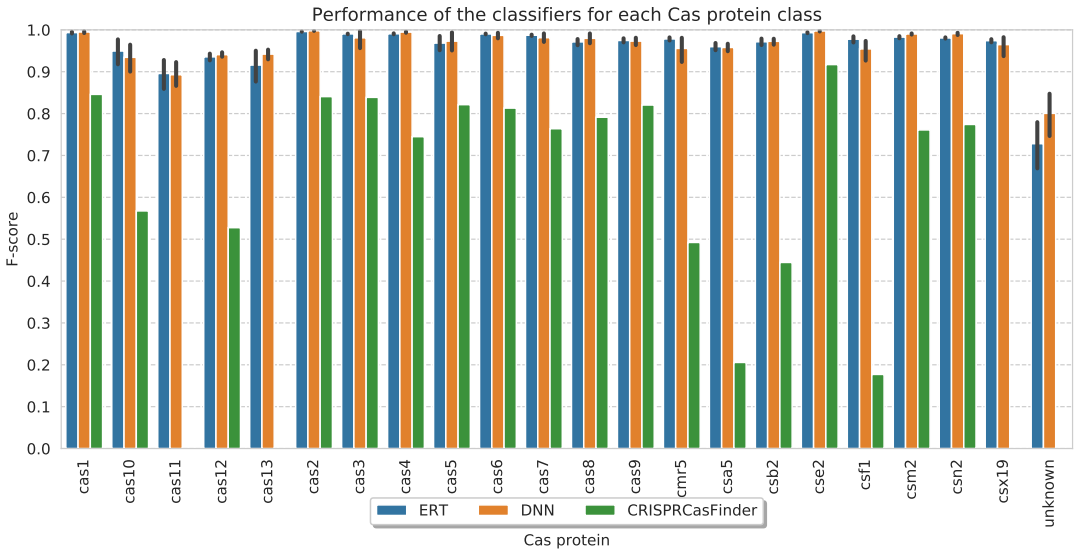
In Figure 19, the average F-scores for Cas type prediction of our method using a combination of specific HMMs and gene properties features are shown. For details of the performance of the models using different types of features, please see our Supplementary Material (Figures 38–46).

Figure 18 – Examples of our method's cassette prediction for the organism Thermotoga sp. RQ2. Specifically, it found two cassettes composed by single interference modules, represented by the orange and green arrows, and a multi-module cassette with two interference modules (blue and red arrows) and an adaptation module (purple arrows). See Figure 37 for more details.



Overall, our method achieved high predictive performances for all Cas types using both ML models. More precisely, for the known Cas types predictions most values are equal to or higher than 0.9. Regarding the prediction of unknown Cas types, ERT and DNN achieved average F-scores of 0.73 and 0.80, respectively. Although the results for unknown Cas types are relatively lower than those of known Cas types, this reduction is expected, given the difficulty of the task for detecting new classes caused by the balancing between the high F-scores for known classes and the ability to potentially point out new genes. The high predictive performance of our models shows their potential for the classification of Cas types for genes in general and for un-predefined *cas* genes observed in many cassettes in particular.

Figure 19 – Comparison of Cas type prediction F-scores between our models (using a combination of the specific HMM and protein properties features) and CRISPRCasFinder. For a comparison between the runtime of Casboundary and CRISPRCasFinder, see Table 32.

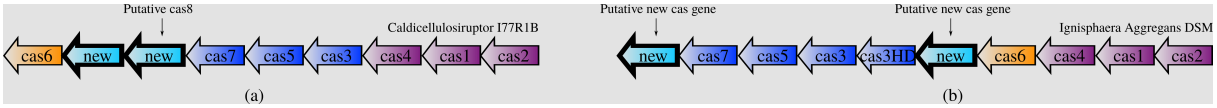


4.5.3 Prediction of potentially new Cas proteins

In this task, we use our method to investigate the problem of predicting (potentially) new Cas proteins, which is a typical scenario for the analysis of novel cassettes. For such we integrated into our method the best ML models that we obtained in the previous section. They are able not only to integrate the knowledge extracted from multiple HMM models and protein properties, but also to generalize the relations among those features.

First, given the cassette boundaries for a genome, we applied our classification methods to label each protein contained in it. Then, we analyzed the proteins that were labeled as "unknown", by performing a clustering search against our database. In Figure 20a, our method labeled two proteins as potentially new. One of them presented a good degree of similarity with a few Cas8 proteins (see our Supplementary Material, Figures 47–49). Since this family is very diverse, this result suggests that it may belong to a new Cas8 subfamily and we labeled the respective gene as "putative cas8". In Figure 20b, our method labeled two genes as potentially new. We did not find any convincing resemblance with the proteins we had in our database. Thus, we believe that such proteins may represent new protein families and we label the respective genes as "putative new *cas* gene".

Figure 20 – Examples of the application of our method for the identification of potentially new Cas proteins, which are marked in bold. In (a), our method predicted two proteins as "new", where one of them has some similarity with Cas8 proteins and may be a new subfamily of Cas8. In (b), our method predicted two proteins as "new", which do not have any similarity to other known Cas proteins and may indicate two new genes.



4.5.4 Occurrence of exchangeable modules

CRISPR cassettes are multi-module structures which are made up of several functional modules each responsible for their own stage of the immune response (VESTERGAARD; GARRETT; SHAH, 2014), including adaptation, processing and interference, in addition of optional accessory modules. The genes comprising each module within a cassette are separated from each other into distinct operons, such that the modules themselves are integral (SHAH *et al.*, 2011). Such a structure enables differential regulation of the expression of the different immune stages, but also enables independent horizontal transfer of a module within a cassette without affecting the functionality of the rest of the immune response. There have been previous reports of CRISPR cassettes from related organisms having undergone such shuffling of modules (GARRETT *et al.*, 2011), although no systematic survey has been made. The capability of our method to define the edges of both Cas modules and cassettes was employed on a database of bacterial and archaeal genomes (Section 4.4.1) and the identities of the detected modules were compared in order to gauge the extent of modular exchange in natural CRISPR-Cas systems.

All cassettes consisting of no more than a single adaptation module and a single interference module were included in the analysis. Adaptation modules from different cassettes were aligned against each other in order to determine their similarity degree. The subtype of each cassette was determined by looking at the interference module. Finally, for each adaptation

module, the subtype of its closest match from a different cassette was recorded in Supplementary Table 31.

Subtypes with a high diagonal percentage close to 100 almost never share their adaptation module with other subtypes of interference modules. I-E and I-F are a good examples of such subtypes, and this observation is consistent with that fact that the adaptation and inferencence stages are coupled in systems of these subtypes, with Cas3 being involved in both stages (WESTRA *et al.*, 2013; VORONTSOVA *et al.*, 2015). On the contrary, and consistent with earlier reports (VESTERGAARD; GARRETT; SHAH, 2014; GARRETT *et al.*, 2011), subtypes I-A, I-B, I-D frequently engage in modular exchange, probably because the adaptation and interference stages are independent in these subtypes (PLAGENS *et al.*, 2012). Besides, most Type III systems have been known for long to piggyback on adaptation and processing machineries of co-occurring Type I systems (HAFT *et al.*, 2005; HALE *et al.*, 2009; MAKAROVA *et al.*, 2011) because they have no such modules of their own, explaining their particularly low diagonal percentages. The extremely low diagonal percentage (37) found for subtype I-U suggests very frequent modular exchange comparable to Type III systems. This result indicates that the subtype co-functions with other CRISPR-Cas systems belonging to subtypes as I-A, I-C, and Type III. This subtype may not have specific adaptation system of its own, like Type III systems. Given that very little experimental data exists on subtype I-U systems, these observations still need confirmation.

4.5.5 Automated annotation of Cas Cassettes and modules

We made our method available as an open source tool in GitHub[1]. It was implemented in Python and is based on the method that integrates our best ML models. Casboundary accepts a complete or partial genome sequence as input, identifies the potential signature genes by using Cas-specific HMM models (MAKAROVA *et al.*, 2019) (see Section 4.4.1), and provides a full identification of the CRISPR cassettes. Next, it labels the genes of the cassette and, as a post-processing step, it can also perform the decomposition of the identified cassette into modules.

Casboundary can be easily integrated with CRISPRcasIdentifier (PADILHA *et al.*, 2020a), a recent tool for the classification of CRISPR cassettes. Casboundary outputs a set of Fasta files containing the identified cassettes, which can be given as input to CRISPRcasIdentifier. As a next step, CRISPRcasIdentifier can classify each cassette into its respective subtype and also predict potentially missing proteins in it. By integrating these tools, the users have a complete CRISPR detection and classification pipeline.

[1] <https://github.com/BackofenLab/Casboundary>

4.6 Conclusion

In this paper, we introduce the first method for automated cassette boundary detection, Cas protein annotation and classification. We apply our method on the datasets from Makarova *et al.* (2015), Makarova *et al.* (2019), which comprise single and multi-module cassettes. Additionally, we also present two real study cases, where we analyze the occurrence of exchangeable models and the prediction of potentially new Cas protein classes.

With respect to boundary detection, the approach followed by our method combines the information available for different genes and a potential signature gene of interest. In our experiments, the method obtains promising predictive performance results as measured by the JS and CL. For single cassettes, we obtain an average JS of 0.86 and CL below 1.09 with the best ML model. For composite cassettes, such a model reaches average JS (resp. CL) values of 0.79 (resp. 1.10) and 0.72 (resp. 1.93) for separated and overlapped cassettes, respectively.

Concerning the Cas protein classification, our method is not only able to assign the Cas type labels for known Cas proteins but also to label a Cas protein as a potentially new type. In our experiments, where we simulate the occurrence of new Cas types by leaving out either 1 or 3 subtypes, our models achieve F-scores above 0.9 for known cas types. Besides, we perform a real study case where our method is able to suggest new putative *cas* genes. Moreover, we conduct another study case to analyze the occurrence of exchangeable models in CRISPR-Cas systems. Our analysis presents evidence of the exchange of adaptation and interference modules in different archea and bacteria CRISPR-Cas systems.

Finally, our method is available as an open source tool in GitHub. At each run, it loads our best ML models and allows the user to apply all the developed methods in an easy and pragmatic way to new CRISPR cassettes.

CHAPTER

5

CONCLUSION

In this thesis, we described and presented experimental results from research investigations and the development and experimental assessment of computational tool in two sub-areas of bioinformatics: (i) application of biclustering coherence measures for the evaluation of biclustering algorithms in gene expression data analysis; and (ii) classification and characterization of CRISPR systems in bacteria and archaea.

Concerning (i), we introduced in Chapter 2 a comparative study of 17 biclustering coherence measures for the evaluation of biclustering results. Specifically, we analyzed when these measures are correlated with external information in the form of gene ontologies for the results of 10 biclustering algorithms that are typically used. A total of 16 different experimental scenarios were investigated which allowed us to show which pairs of measures are redundant in which scenarios. Furthermore, with the time complexity analyses, we were able to recommend the measures that require less computational effort, which is an important topic to consider for large-scale applications. Finally, we observed that no coherence measure presented strong correlations with the information from the gene ontologies. Thus, combining both types of evaluation may improve the relevance of the experimental results in an analysis of gene expression using biclustering algorithms.

With respect to (ii), we presented two experimental studies and tool developments, in Chapters 3 and 4. In Chapter 3, we introduced CRISPRcasIdentifier, a computational tool for classifying CRISPR cassettes. This new tool was designed in agreement with the most recent CRISPR classification studies of Makarova *et al.* (2015), Makarova *et al.* (2019). In our experiments, CRISPRcasIdentifier presented very high accuracy results, above 0.95, and also achieved better results than other tools on the last published CRISPR dataset (MAKAROVA *et al.*, 2019). Specifically, CRISPRcasIdentifier reached an adjusted balanced accuracy of 0.89 and a F-score with macro averaging of 0.91 against 0.54 and 0.63 for Macsyfinder and CRISPRdisco, which are currently the best competitors, respectively. Moreover, CRISPRcasIdentifier is the first tool that can estimate the normalized bit-score of potentially missing proteins in CRISPR

cassettes. In the correspondent experiments, CRISPRcasIdentifier obtained, in general, mean absolute errors below 0.05.

In Chapter 4, we presented Casboundary, which is the first tool that defines the boundaries of the cassettes of archaeal and bacterial genomes. Specifically, it takes into account the relation between potential signatures and their neighboring genes to define the CRISPR cassettes. It also allows the decomposition of cassettes into modules. Moreover, it performs the classification of core and signature genes and predicts potentially new *cas* gene types. Finally, we presented a study case that illustrated the application of our tool in a real scenario, where we reported putative new *cas* gene types.

5.1 Limitations and future work

Most of the biclustering coherence measures that were analyzed in this thesis can be easily influenced by the bicluster size. In particular, it is known that biclusters containing few rows and/or columns may present coherent patterns only due to chance (HENRIQUES; MADEIRA, 2018). Recently, Iorio, Chiaromonte and Cremona (2020) discussed this bias for the MSR measure and proposed a simple normalization scheme for its correction, supported by theoretical proofs. It might be relevant to investigate how to extend their study for other frequently used measures, such as VE and VEt, that are able to identify more patterns than the MSR.

Another possibility is to investigate on how to empirically adjust some coherence measures for chance or to estimate p-values that indicate the significance of their experimental results. Although this approach is computationally expensive when compared to the approach investigated in Iorio, Chiaromonte and Cremona (2020), it has the advantage of being easily applicable to many coherence measures with minor modifications. Some studies have already proposed sampling procedures for problems such as graph mining (HANHIJÄRVI; GARRIGA; PUOLAMÄKI, 2009), clustering (OJALA *et al.*, 2009), or frequent itemset mining (HANHIJÄRVI *et al.*, 2009). Those methods usually perform some perturbation to the original dataset, but without modifying some of its core characteristics, such as the distributions of row or column values. We believe that these additional studies may provide interesting starting points for the investigation of experimental chance adjustment or p-value estimation for biclustering.

CRISPRcasIdentifier is not able to identify potentially new CRISPR subtypes. Casboundary can indicate potentially new Cas proteins, but it is not capable of discriminating unseen classes. Hence, the application of methods able to learn potential new classes formed by large sets of putative new cassette subtypes or protein families could provide important research contributions. Recently, zero-shot learning methods have been developed and mainly applied to image classification datasets. These approaches assume that not all classes are available during training. Thus, they try to transfer the knowledge obtained in this step to learn new classes in the

test phase (WANG *et al.*, 2019). For such, besides the original feature space of the problem, it is also necessary another space, called *semantic space*. Semantic spaces are expected to summarize general information of the seen and unseen classes as different prototypes, which consist of feature vectors in this space.

We believe that it may be also be relevant to investigate the adaption or development of new zero-shot learning techniques for CRISPR-Cas systems. As a starting point, the features that have been used in Chapters 3 and 4 could be investigated as the original feature space. The semantic space could be defined by extra gene and protein-related features, such as those implemented in the packages protr (XIAO *et al.*, 2015), iFeature (CHEN *et al.*, 2018) and PyFeat (MUHAMMOD *et al.*, 2019).

Zero-shot learning may fit well for the classification of protein families and cassettes of CRISPR-Cas systems. Furthermore, a new tool could speed up analyses with new archaeal and bacterial genomes and help in the new classification studies that are periodically published , especially considering the fast evolutionary characteristic of these systems.

BIBLIOGRAPHY

ABBY, S. S. *et al.* Macsyfinder: A program to mine genomes for molecular systems with an application to crispr-cas systems. **PLOS ONE**, Public Library of Science, v. 9, n. 10, p. 1–9, 10 2014. Available: <https://doi.org/10.1371/journal.pone.0110726>. Citations on pages 67 and 82.

AGUILAR-RUIZ, J. S. Shifting and scaling patterns from gene expression data. **Bioinformatics**, Oxford University Press, v. 21, n. 20, p. 3840–3845, 2005. Citation on page 44.

ALKHNBASHI, O. S. **Computational Characterisation of Genomic CRISPR-Cas Systems in Archaea and Bacteria**. Phd Thesis (PhD Thesis) — Albert-Ludwigs-Universität Freiburg, 2017. Citations on pages 28, 33, and 34.

ALKHNBASHI, O. S.; COSTA, F.; SHAH, S. A.; GARRETT, R. A.; SAUNDERS, S. J.; BACKOFEN, R. CRISPRstrand: Predicting repeat orientations to determine the crRNA-encoding strand at CRISPR loci. **Bioinformatics (Oxford, England)**, v. 30, n. 17, p. i489–496, Sep. 2014. ISSN 1367-4811. Citations on pages 67 and 87.

ALKHNBASHI, O. S. *et al.* Characterizing leader sequences of crispr loci. **Bioinformatics**, v. 32, n. 17, p. i576–i585, 2016. Citations on pages 67, 68, and 87.

____. Crispr-cas bioinformatics. **Methods**, p. 3–11, 02 2020. Citation on page 87.

ARSLAN, Z. *et al.* Double-strand DNA end-binding and sliding of the toroidal CRISPR-associated protein Csn2. **Nucleic Acids Research**, v. 41, n. 12, p. 6347–6359, 04 2013. ISSN 0305-1048. Citation on page 79.

ASHBURNER, M.; BALL, C. A.; BLAKE, J. A.; BOTSTEIN, D.; BUTLER, H.; CHERRY, J. M.; DAVIS, A. P.; DOLINSKI, K.; DWIGHT, S. S.; EPPIG, J. T. *et al.* Gene ontology: tool for the unification of biology. **Nature genetics**, Nature Publishing Group, v. 25, n. 1, p. 25, 2000. Citations on pages 32 and 56.

AYADI, W.; ELLOUMI, M.; HAO, J.-K. A biclustering algorithm based on a bicluster enumeration tree: application to dna microarray data. **BioData mining**, BioMed Central, v. 2, n. 1, p. 9, 2009. Citations on pages 50 and 53.

BATEMAN, A.; COIN, L.; DURBIN, R.; FINN, R. D.; HOLLICH, V.; GRIFFITHS-JONES, S.; KHANNA, A.; MARSHALL, M.; MOXON, S.; SONNHAMMER, E. L. *et al.* The pfam protein families database. **Nucleic acids research**, Oxford University Press, v. 32, n. suppl_1, p. D138–D141, 2004. Citation on page 92.

BEN-DOR, A.; CHOR, B.; KARP, R.; YAKHINI, Z. Discovering local structure in gene expression data: the order-preserving submatrix problem. **Journal of computational biology**, Mary Ann Liebert, Inc., v. 10, n. 3-4, p. 373–384, 2003. Citation on page 54.

BEN-DOR, A.; SHAMIR, R.; YAKHINI, Z. Clustering gene expression patterns. **Journal of computational biology**, Mary Ann Liebert, Inc., v. 6, n. 3-4, p. 281–297, 1999. Citation on page 42.

BISHOP, C. M. **Pattern recognition and machine learning**. [S.l.]: springer, 2006. Citations on pages 70 and 75.

BISWAS, A.; STAALS, R. H. J.; MORALES, S. E.; FINERAN, P. C.; BROWN, C. M. Crisprdetect: A flexible algorithm to define crispr arrays. **BMC Genomics**, p. i356–i356, 2016. Citation on page 67.

BOZDAĞ, D.; KUMAR, A. S.; CATALYUREK, U. V. Comparative analysis of biclustering algorithms. In: ACM. **Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology**. [S.l.], 2010. p. 265–274. Citation on page 43.

BREIMAN, L. *et al.* **Classification and regression trees**. [S.l.]: Chapman & Hall/CRC, 1984. Citation on page 69.

BRODERSEN, K. H.; ONG, C. S.; STEPHAN, K. E.; BUHMANN, J. M. The balanced accuracy and its posterior distribution. In: IEEE. **2010 20th International Conference on Pattern Recognition**. [S.l.], 2010. p. 3121–3124. Citation on page 70.

BROWN, P. O.; BOTSTEIN, D. Exploring the new world of the genome with dna microarrays. **Nature genetics**, Nature Publishing Group, v. 21, n. 1s, p. 33, 1999. Citations on pages 30 and 42.

BUSYGIN, S.; PROKOPYEV, O.; PARDALOS, P. M. Biclustering in data mining. **Computers & Operations Research**, Elsevier, v. 35, n. 9, p. 2964–2987, 2008. Citation on page 43.

CASS, S. D. B.; HAAS, K. A.; STOLL, B.; ALKHNBASHI, O.; SHARMA, K.; URLAUB, H.; BACKOFEN, R.; MARCHFELDER, A.; BOLT, E. L. The role of Cas8 in type I CRISPR interference. **Biosci Rep**, v. 35, n. 4, p. e00197, 2015. ISSN 0144-8463. Citation on page 66.

CAWLEY, G. C.; TALBOT, N. L. On over-fitting in model selection and subsequent selection bias in performance evaluation. **Journal of Machine Learning Research**, v. 11, n. Jul, p. 2079–2107, 2010. Citation on page 70.

CHAI, G.; YU, M.; JIANG, L.; DUAN, Y.; HUANG, J. Hmmcas: a web tool for the identification and domain annotations of cas proteins. **IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)**, IEEE Computer Society Press, v. 16, n. 4, p. 1313–1315, 2019. Citations on pages 67 and 82.

CHEN, S.; LIU, J.; ZENG, T. Measuring the quality of linear patterns in biclusters. **Methods**, Elsevier, v. 83, p. 18–27, 2015. Citations on pages 44, 46, 49, and 53.

CHEN, Z.; ZHAO, P.; LI, F.; LEIER, A.; MARQUEZ-LAGO, T. T.; WANG, Y.; WEBB, G. I.; SMITH, A. I.; DALY, R. J.; CHOU, K.-C. *et al.* ifeature: a python package and web server for features extraction and selection from protein and peptide sequences. **Bioinformatics**, Oxford University Press, v. 34, n. 14, p. 2499–2502, 2018. Citation on page 103.

CHENG, Y.; CHURCH, G. M. Biclustering of expression data. In: **ISMB**. [S.l.]: AAAI Press, 2000. v. 8, p. 93–103. Citations on pages 31, 42, 44, 47, 53, 54, 56, and 58.

CHERKASSKY, V.; DHAR, S. Simple method for interpretation of high-dimensional nonlinear svm classification models. In: **6th International Conference on Data Mining**. [S.l.: s.n.], 2010. p. 267–272. Citations on pages 17, 76, and 124.

CHRISTIE, K. R.; HONG, E. L.; CHERRY, J. M. Functional annotations for the saccharomyces cerevisiae genome: the knowns and the known unknowns. **Trends in microbiology**, Elsevier, v. 17, n. 7, p. 286–294, 2009. Citation on page 56.

CHU, S.; DERISI, J.; EISEN, M.; MULHOLLAND, J.; BOTSTEIN, D.; BROWN, P. O.; HERSKOWITZ, I. The transcriptional program of sporulation in budding yeast. **Science**, American Association for the Advancement of Science, v. 282, n. 5389, p. 699–705, 1998. Citation on page 56.

CONSORTIUM, G. O. The gene ontology (go) database and informatics resource. **Nucleic acids research**, Oxford University Press, v. 32, n. suppl_1, p. D258–D261, 2004. Citations on pages 32 and 56.

COUVIN, D.; BERNHEIM, A.; Toffano-Nioche, C.; TOUCHON, M.; MICHALIK, J.; NÉRON, B.; ROCHA, E. P. C.; VERGNAUD, G.; GAUTHERET, D.; POURCEL, C. CRISPRCasFinder, an update of CRISRFinder, includes a portable version, enhanced performance and integrates search for Cas proteins. **Nucleic Acids Research**, v. 46, n. W1, p. W246–W251, Jul. 2018. ISSN 0305-1048. Citations on pages 67, 82, 87, and 95.

CRAWLEY, A. B. *et al.* CRISPRdisco: An Automated Pipeline for the Discovery and Analysis of CRISPR-Cas Systems. **The CRISPR Journal**, v. 1, n. 2, p. 171–181, Apr. 2018. ISSN 2573-1599. Citations on pages 67, 82, and 87.

DENG, L. *et al.* Modulation of CRISPR locus transcription by the repeat-binding protein Cbp1 in Sulfolobus. **NAR**, v. 40, n. 6, p. 2470–80, 2012. ISSN 0305-1048. Citation on page 67.

DIVINA, F.; PONTES, B.; GIRÁLDEZ, R.; AGUILAR-RUIZ, J. S. An effective measure for assessing the quality of biclusters. **Computers in biology and medicine**, Elsevier, v. 42, n. 2, p. 245–256, 2012. Citations on pages 44, 52, and 53.

EBINA, H.; MISAWA, N.; KANEMURA, Y.; KOYANAGI, Y. Harnessing the crispr/cas9 system to disrupt latent hiv-1 provirus. **Scientific reports**, Nature Publishing Group, v. 3, p. 2510, 2013. Citation on page 36.

EDGAR, R. C. MUSCLE: multiple sequence alignment with high accuracy and high throughput. **Nucleic Acids Research**, v. 32, p. 1792–1797, Mar. 2004. ISSN 5. Citation on page 68.

ENRIGHT, A. J. *et al.* An efficient algorithm for large-scale detection of protein families. **Nucleic Acids Research**, v. 30, n. 7, p. 1575–1584, Apr. 2002. Citations on pages 68 and 72.

EREN, K.; DEVECI, M.; KÜÇÜKTUNÇ, O.; ÇATALYÜREK, Ü. V. A comparative analysis of biclustering algorithms for gene expression data. **Briefings in bioinformatics**, Oxford University Press, v. 14, n. 3, p. 279–292, 2012. Citations on pages 43 and 57.

FINN, R. D.; CLEMENTS, J.; EDDY, S. R. Hmmer web server: interactive sequence similarity searching. **Nucleic acids research**, Oxford University Press, v. 39, n. suppl_2, p. W29–W37, 2011. Citations on pages 67 and 68.

FISHER, R. A. On the interpretation of χ^2 from contingency tables, and the calculation of p. **Journal of the Royal Statistical Society**, JSTOR, v. 85, n. 1, p. 87–94, 1922. Citation on page 32.

FLORES, J. L.; INZA, I.; LARRAÑAGA, P.; CALVO, B. A new measure for gene expression biclustering based on non-parametric correlation. **Computer methods and programs in biomedicine**, Elsevier, v. 112, n. 3, p. 367–397, 2013. Citations on pages 44, 51, and 53.

FORMAN, G.; SCHOLZ, M. Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement. **ACM SIGKDD Explorations Newsletter**, ACM, v. 12, n. 1, p. 49–57, 2010. Citation on page 70.

GARNEAU, J. E. o. The CRISPR/Cas bacterial immune system cleaves bacteriophage and plasmid DNA. **Nature**, v. 468, n. 7320, p. 67–71, 2010. ISSN 1476-4687. Citation on page 66.

GARRETT, R. A. *et al.* Archaeal CRISPR-based immune systems: exchangeable functional modules. **Trends Microbiol**, v. 19, n. 11, p. 549–56, 2011. ISSN 1878-4380. Citations on pages 87, 91, 98, and 99.

GASCH, A. P.; SPELLMAN, P. T.; KAO, C. M.; CARMEL-HAREL, O.; EISEN, M. B.; STORZ, G.; BOTSTEIN, D.; BROWN, P. O. Genomic expression programs in the response of yeast cells to environmental changes. **Molecular biology of the cell**, Am Soc Cell Biol, v. 11, n. 12, p. 4241–4257, 2000. Citation on page 56.

GEURTS, P. *et al.* Extremely randomized trees. **Machine learning**, Springer, v. 63, n. 1, p. 3–42, 2006. Citations on pages 70 and 93.

GIRALDEZ, R.; DIVINA, F.; PONTES, B.; AGUILAR-RUIZ, J. S. Evolutionary search of biclusters by minimal intrafluctuation. In: IEEE. **FUZZ-IEEE**. [S.l.], 2007. p. 1–6. Citations on pages 44, 52, and 53.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Citation on page 93.

GU, J.; LIU, J. S. Bayesian biclustering of gene expression data. **BMC genomics**, BioMed Central, v. 9, n. 1, p. S4, 2008. Citation on page 55.

GUYON, I.; BENNETT, K.; CAWLEY, G.; ESCALANTE, H. J.; ESCALERA, S.; HO, T. K.; MACIA, N.; RAY, B.; SAEED, M.; STATNIKOV, A.; VIEGAS, E. Design of the 2015 chalearn automl challenge. In: IEEE. **2015 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2015. p. 1–8. Citation on page 70.

HAFT, D. H. *et al.* A guild of 45 CRISPR-associated (Cas) protein families and multiple CRISPR/Cas subtypes exist in prokaryotic genomes. **PLoS Comput Biol**, v. 1, n. 6, p. e60, 2005. Citations on pages 35, 67, and 99.

HAFT, D. H.; SELENGUT, J. D.; WHITE, O. The tigrfams database of protein families. **Nucleic acids research**, Oxford University Press, v. 31, n. 1, p. 371–373, 2003. Citation on page 92.

HALE, C. R. *et al.* Rna-guided rna cleavage by a crispr rna-cas protein complex. **Cell**, p. 945–956, 11 2009. Citation on page 99.

HANHIJÄRVI, S.; GARRIGA, G. C.; PUOLAMÄKI, K. Randomization techniques for graphs. In: SIAM. **Proceedings of the 2009 SIAM International Conference on Data Mining**. [S.l.], 2009. p. 780–791. Citation on page 102.

HANHIJÄRVI, S.; OJALA, M.; VUOKKO, N.; PUOLAMÄKI, K.; TATTI, N.; MANNILA, H. Tell me something i don't know: randomization strategies for iterative data mining. In: **Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2009. p. 379–388. Citation on page 102.

HARTIGAN, J. A. Direct clustering of a data matrix. **Journal of the american statistical association**, Taylor & Francis Group, v. 67, n. 337, p. 123–129, 1972. Citations on pages 31, 47, and 53.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference and Prediction**. 2. ed. [S.l.]: Springer, 2009. Citation on page 70.

HE, F.; VESTERGAARD, G.; PENG, W.; SHE, Q.; PENG, X. Crispr-cas type i-a cascade complex couples viral infection surveillance to host transcriptional regulation in the dependence of csa3b. **Nucleic Acids Research**, Published by Oxford University Press on behalf of Nucleic Acids Research, 2 2017. Citation on page 79.

HENRIQUES, R.; MADEIRA, S. C. Bsig: evaluating the statistical significance of biclustering solutions. **Data Mining and Knowledge Discovery**, Springer, v. 32, n. 1, p. 124–161, 2018. Citation on page 102.

HOCHBERG, Y.; BENJAMINI, Y. More powerful procedures for multiple significance testing. **Statistics in medicine**, Wiley Online Library, v. 9, n. 7, p. 811–818, 1990. Citations on pages 32 and 57.

HOCHREITER, S.; BODENHOFER, U.; HEUSEL, M.; MAYR, A.; MITTERECKER, A.; KASIM, A.; KHAMIAKOVA, T.; SANDEN, S. V.; LIN, D.; TALLOEN, W. *et al.* Fabia: factor analysis for bicluster acquisition. **Bioinformatics**, Oxford University Press, v. 26, n. 12, p. 1520–1527, 2010. Citation on page 55.

HORTA, D.; CAMPELLO, R. J. G. B. Similarity measures for comparing biclusterings. **IEEE/ACM TCBB**, IEEE, v. 11, n. 5, p. 942–954, 2014. Citation on page 58.

HORVATH, P.; ROMERO, D. A.; COÛTÉ-MONVOISIN, A.-C.; RICHARDS, M.; DEVEAU, H.; MOINEAU, S.; BOYAVAL, P.; FREMAUX, C.; BARRANGOU, R. Diversity, activity, and evolution of crispr loci in streptococcus thermophilus. **Journal of Bacteriology**, American Society for Microbiology Journals, v. 190, n. 4, p. 1401–1412, 2008. ISSN 0021-9193. Available: <https://jb.asm.org/content/190/4/1401>. Citation on page 86.

HSU, C.-W.; CHANG, C.-C.; LIN, C.-J. *et al.* A practical guide to support vector classification. Taipei, Taiwan, 2003. Citation on page 71.

HSU, P. D.; LANDER, E. S.; ZHANG, F. Development and applications of crispr-cas9 for genome engineering. **Cell**, Elsevier, v. 157, n. 6, p. 1262–1278, 2014. Citation on page 36.

HYATT, D.; CHEN, G.-L.; LOCASCIO, P. F.; LAND, M. L.; LARIMER, F. W.; HAUSER, L. J. Prodigal: prokaryotic gene recognition and translation initiation site identification. **BMC Bioinformatics**, BMC Bioinformatics, 03 2010. Citations on pages 69 and 91.

IORIO, J. D.; CHIAROMONTE, F.; CREMONA, M. A. On the bias of h-scores for comparing biclusters, and how to correct it. **Bioinformatics**, Oxford University Press, v. 36, n. 9, p. 2955–2957, 2020. Citation on page 102.

JAIN, A. K.; DUBES, R. C. Algorithms for clustering data. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1988. Citation on page 30.

JANSEN, R.; EMBDEN, J. D. v.; GAASTRA, W.; SCHOULS, L. M. Identification of genes that are associated with dna repeats in prokaryotes. **Molecular microbiology**, Wiley Online Library, v. 43, n. 6, p. 1565–1575, 2002. Citation on page 35.

JASKOWIAK, P. A.; CAMPELLO, R. J.; COSTA, I. G. On the selection of appropriate distances for gene expression data clustering. In: BIOMED CENTRAL. **BMC bioinformatics**. [S.l.], 2014. v. 15, n. 2, p. S2. Citation on page 56.

JASKOWIAK, P. A.; CAMPELLO, R. J. G. B.; COSTA, I. G. Proximity measures for clustering gene expression microarray data: a validation methodology and a comparative analysis. **IEEE/ACM TCBB**, IEEE, v. 10, n. 4, p. 845–857, 2013. Citation on page 56.

JIANG, D.; TANG, C.; ZHANG, A. Cluster analysis for gene expression data: A survey. **IEEE TKDE**, IEEE, v. 16, n. 11, p. 1370–1386, 2004. Citations on pages 31 and 42.

KHATODIA, S.; BHATOTIA, K.; PASSRICHA, N.; KHURANA, S.; TUTEJA, N. The crispr/cas genome-editing tool: application in improvement of crops. **Frontiers in plant science**, Frontiers, v. 7, p. 506, 2016. Citation on page 37.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. In: **Proceedings of the 3rd International Conference on Learning Representations**. San Diego, CA, USA: [s.n.], 2015. Citation on page 94.

KLUGER, Y.; BASRI, R.; CHANG, J. T.; GERSTEIN, M. Spectral biclustering of microarray data: coclustering genes and conditions. **Genome research**, Cold Spring Harbor Lab, v. 13, n. 4, p. 703–716, 2003. Citation on page 54.

KOO, Y. *et al.* Crystal structure of streptococcus pyogenes csn2 reveals calcium-dependent conformational changes in its tertiary and quaternary structure. **PLOS ONE**, Public Library of Science, v. 7, n. 3, p. 1–8, 03 2012. Citation on page 79.

KOONIN, E. V.; MAKAROVA, K. S.; WOLF, Y. I.; KRUPOVIC, M. Evolutionary entanglement of mobile genetic elements and host defence systems: guns for hire. **Nature Reviews Microbiology**, Nature Publishing Group, p. 1–17, 2020. Citation on page 86.

KRIEGEL, H.-P.; KRÖGER, P.; ZIMEK, A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. **ACM Transactions on Knowledge Discovery from Data (TKDD)**, ACM, v. 3, n. 1, p. 1, 2009. Citation on page 31.

KRSTAJIC, D. *et al.* Cross-validation pitfalls when selecting and assessing regression and classification models. **Journal of cheminformatics**, Nature Publishing Group, v. 6, n. 1, p. 10, 2014. Citation on page 70.

LANGE, S. J.; ALKHNBASHI, O. S.; ROSE, D.; WILL, S.; BACKOFEN, R. CRISPRmap: An automated classification of repeat conservation in prokaryotic adaptive immune systems. **Nucleic Acids Research**, v. 41, n. 17, p. 8034–8044, Jan. 2013. ISSN 0305-1048, 1362-4962. Citations on pages 67 and 87.

LEE, K.-H. *et al.* Identification, structural, and biochemical characterization of a group of large csn2 proteins involved in crispr-mediated bacterial immunity. **Proteins**, v. 80 11, p. 2573–82, 2012. Citation on page 79.

LI, G.; MA, Q.; TANG, H.; PATERSON, A. H.; XU, Y. Qubic: a qualitative biclustering algorithm for analyses of gene expression data. **Nucleic acids research**, Oxford University Press, v. 37, n. 15, p. e101–e101, 2009. Citations on pages 55 and 57.

LIU, W.; WANG, Z.; LIU, X.; ZENG, N.; LIU, Y.; ALSAADI, F. E. A survey of deep neural network architectures and their applications. **Neurocomputing**, Elsevier, v. 234, p. 11–26, 2017. Citation on page 93.

LODISH, H.; BERK, A.; MATSUDAIRA, P.; KAISER, C. A.; KRIEGER, M.; SCOTT, M. P.; ZIPURSKY, L.; DARNELL, J. **Molecular cell biology**. 5. ed. [S.l.]: Macmillan, 2008. Citations on pages 28 and 30.

LUSCOMBE, N. M.; GREENBAUM, D.; GERSTEIN, M. What is bioinformatics? a proposed definition and overview of the field. **Methods of information in medicine**, Schattauer GmbH, v. 40, n. 04, p. 346–358, 2001. Citation on page 27.

MADEIRA, S. C.; OLIVEIRA, A. L. Biclustering algorithms for biological data analysis: a survey. **IEEE/ACM TCBB**, IEEE Computer Society Press, v. 1, n. 1, p. 24–45, 2004. Citations on pages 30, 31, 42, 43, and 46.

MAKAROVA, K. S.; HAFT, D. H.; BARRANGOU, R.; BROUNS, S. J. J.; CHARPENTIER, E.; HORVATH, P.; MOINEAU, S.; MOJICA, F. J. M.; WOLF, Y. I.; YAKUNIN, A. F.; OOST, J. v. d.; KOONIN, E. V. Evolution and classification of the CRISPR-Cas systems. **Nature Reviews Microbiology**, v. 9, n. 6, p. 467–77, 2011. ISSN 1740-1526. Citations on pages 35, 67, and 99.

MAKAROVA, K. S.; WOLF, Y. I.; ALKHNBASHI, O. S.; COSTA, F.; SHAH, S. A.; SAUNDERS, S. J.; BARRANGOU, R.; BROUNS, S. J. J.; CHARPENTIER, E.; HAFT, D. H.; HORVATH, P.; MOINEAU, S.; MOJICA, F. J. M.; TERNS, R. M.; TERNS, M. P.; WHITE, M. F.; YAKUNIN, A. F.; GARRETT, R. A.; van der Oost, J.; BACKOFEN, R.; KOONIN, E. V. An updated evolutionary classification of CRISPR-Cas systems. **Nature Reviews Microbiology**, v. 13, n. 11, p. 722–736, Nov. 2015. ISSN 1740-1526. Citations on pages 35, 36, 38, 66, 67, 68, 69, 71, 74, 76, 77, 79, 83, 84, 87, 88, 91, 92, 100, 101, and 120.

MAKAROVA, K. S.; WOLF, Y. I.; IRANZO, J.; SHMAKOV, S.; ALKHNBASHI, O. S.; COSTA, F.; SHAH, S. A.; SAUNDERS, S. J.; BARRANGOU, R.; BROUNS, S. J. J.; CHARPENTIER, E.; HAFT, D. H.; HORVATH, P.; MOINEAU, S.; MOJICA, F. J. M.; TERNS, R. M.; TERNS, M. P.; WHITE, M. F.; YAKUNIN, A. F.; GARRETT, R. A.; van der Oost, J.; BACKOFEN, R.; KOONIN, E. V. Evolutionary classification of crispr–cas systems: a burst of class 2 and derived variants. **Nature Reviews Microbiology**, Nature Publishing Group, p. 1–17, 2019. Citations on pages 33, 35, 36, 38, 82, 84, 86, 88, 91, 92, 99, 100, and 101.

MANNING, C.; RAGHAVAN, P.; SCHÜTZE, H. **Introduction to information retrieval**. [S.l.]: Cambridge University Press, 2009. Citation on page 92.

MARCHLER-BAUER, A.; BRYANT, S. H. CD-Search: protein domain annotations on the fly. **Nucleic Acids Research**, v. 32, n. suppl_2, p. W327–W331, 07 2004. ISSN 0305-1048. Available: <https://doi.org/10.1093/nar/gkh454>. Citation on page 67.

MARCHLER-BAUER, A.; LU, S.; ANDERSON, J. B.; CHITSAZ, F.; DERBYSHIRE, M. K.; DEWEESE-SCOTT, C.; FONG, J. H.; GEER, L. Y.; GEER, R. C.; GONZALES, N. R. *et al.* Cdd: a conserved domain database for the functional annotation of proteins. **Nucleic acids research**, Oxford University Press, v. 39, n. suppl_1, p. D225–D229, 2010. Citation on page 92.

MOJICA, F. J.; GARRETT, R. A. Discovery and seminal developments in the crispr field. In: **CRISPR-Cas Systems**. [S.l.]: Springer, 2013. p. 1–31. Citation on page 34.

MUHAMMOD, R.; AHMED, S.; FARID, D. M.; SHATABDA, S.; SHARMA, A.; DEHZANGI, A. Pyfeat: a python-based effective feature generation tool for dna, rna and protein sequences. **Bioinformatics**, Oxford University Press, v. 35, n. 19, p. 3831–3833, 2019. Citation on page 103.

MUKHOPADHYAY, A.; MAULIK, U.; BANDYOPADHYAY, S. A novel coherence measure for discovering scaling biclusters from gene expression data. **Journal of bioinformatics and computational biology**, World Scientific, v. 7, n. 05, p. 853–868, 2009. Citations on pages 44, 49, and 53.

NAM, K. H. *et al.* Crystal structure of clustered regularly interspaced short palindromic repeats (CRISPR)-associated Csn2 protein revealed Ca2+-dependent double-stranded DNA binding activity. **J Biol Chem**, v. 35, p. 30759–68, 09 2011. ISSN 0305-1048. Citation on page 79.

NEPOMUCENO, J. A.; TRONCOSO, A.; AGUILAR-RUIZ, J. S. Biclustering of gene expression data by correlation-based scatter search. **BioData mining**, BioMed Central, v. 4, n. 1, p. 3, 2011. Citations on pages 44, 49, and 53.

OGHABIAN, A.; KILPINEN, S.; HAUTANIEMI, S.; CZEIZLER, E. Biclustering methods: biological relevance and application in gene expression analysis. **PloS one**, Public Library of Science, v. 9, n. 3, p. e90801, 2014. Citation on page 43.

OJALA, M.; VUOKKO, N.; KALLIO, A.; HAIMINEN, N.; MANNILA, H. Randomization methods for assessing data analysis results on real-valued matrices. **Statistical Analysis and Data Mining: The ASA Data Science Journal**, Wiley Online Library, v. 2, n. 4, p. 209–230, 2009. Citation on page 102.

PADILHA, V. A.; ALKHNBASHI, O. S.; SHAH, S. A.; DE CARVALHO, A. C. P. L. F.; BACKOFEN, R. CRISPRcasIdentifier: Machine learning for accurate identification and classification of CRISPR-Cas systems. **GigaScience**, v. 9, n. 6, 06 2020. ISSN 2047-217X. Giaa062. Available: <https://doi.org/10.1093/gigascience/giaa062>. Citations on pages 38 and 99.

PADILHA, V. A.; ALKHNBASHI, O. S.; SHAH, S. A.; CARVALHO, A. C. P. L. F.; BACKOFEN, R. Supporting data for "crisprcasidentifier: Machine learning for accurate identification and classification of crispr-cas systems". **GigaScience Database**, 2020. Available: <http://dx.doi.org/10.5524/100751>. Citation on page 84.

PADILHA, V. A.; ALKHNBASHI, O. S.; TRAN, V. D.; SHAH, S. A.; CARVALHO, A. C. P. L. F.; BACKOFEN, R. Casboundary: Automated definition of integral Cas cassettes. **Bioinformatics**, 2020. ISSN 1367-4803. Btaa984. Available: <https://doi.org/10.1093/bioinformatics/btaa984>. Citation on page 39.

PADILHA, V. A.; CAMPELLO, R. J. G. B. A systematic comparative evaluation of biclustering techniques. **BMC bioinformatics**, BioMed Central, v. 18, n. 1, p. 55, 2017. Citations on pages 31, 43, 55, 56, and 57.

PADILHA, V. A.; CARVALHO, A. C. P. L. F. A study of biclustering coherence measures for gene expression data. In: IEEE. **7th Brazilian Conference on Intelligent Systems**. [S.l.], 2018. p. 546–551. Citations on pages 38 and 43.

____. Experimental correlation analysis of bicluster coherence measures and gene ontology information. **Applied Soft Computing**, Elsevier, v. 85, p. 105688, 2019. Available: <https://doi.org/10.1016/j.asoc.2019.105688>. Citation on page 38.

PEARSON, W. R.; LIPMAN, D. J. Improved tools for biological sequence comparison. **Proceedings of the National Academy of Sciences**, National Academy of Sciences, v. 85, n. 8, p. 2444–2448, 1988. ISSN 0027-8424. Available: <https://www.pnas.org/content/85/8/2444>. Citation on page 68.

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.; DUCHESNAY, E. Scikit-learn: Machine learning in python. **Journal of machine learning research**, v. 12, n. Oct, p. 2825–2830, 2011. Citation on page 71.

PLAGENS, A. *et al.* Characterization of the crispr/cas subtype i-a system of the hyperthermophilic crenarchaeon thermoproteus tenax. **J Bacteriol**, p. 2491–2500, 05 2012. Citation on page 99.

PONTES, B.; GIRÁLDEZ, R.; AGUILAR-RUIZ, J. S. Measuring the quality of shifting and scaling patterns in biclusters. In: SPRINGER. **IAPR PRIB**. [S.l.], 2010. p. 242–252. Citations on pages 44, 52, and 53.

____. Biclustering on expression data: A review. **Journal of biomedical informatics**, Elsevier, v. 57, p. 163–180, 2015. Citations on pages 31, 43, and 45.

PONTES, B.; GIRLDEZ, R.; AGUILAR-RUIZ, J. S. Quality measures for gene expression biclusters. **PloS one**, Public Library of Science, v. 10, n. 3, p. e0115497, 2015. Citations on pages 42, 43, 44, 46, 49, 51, 52, 57, 62, and 63.

PRELIĆ, A.; BLEULER, S.; ZIMMERMANN, P.; WILLE, A.; BÜHLMANN, P.; GRUISSEM, W.; HENNIG, L.; THIELE, L.; ZITZLER, E. A systematic comparison and evaluation of biclustering methods for gene expression data. **Bioinformatics**, Oxford University Press, v. 22, n. 9, p. 1122–1129, 2006. Citations on pages 43, 54, 56, and 57.

RAN, F. A.; HSU, P. D.; WRIGHT, J.; AGARWALA, V.; SCOTT, D. A.; ZHANG, F. Genome engineering using the crispr-cas9 system. **Nature protocols**, Nature Publishing Group, v. 8, n. 11, p. 2281, 2013. Citation on page 36.

RATH, D.; AMLINGER, L.; RATH, A.; LUNDGREN, M. The crispr-cas immune system: biology, mechanisms and applications. **Biochimie**, Elsevier, v. 117, p. 119–128, 2015. Citations on pages 33 and 34.

REMMERT, M. *et al.* Hhblits: lightning-fast iterative protein sequence searching by hmm-hmm alignment. **Nature Methods**, v. 9, p. 173–175, 02 2012. Citations on pages 16, 81, and 87.

RHODES, D. R.; YU, J.; SHANKER, K.; DESHPANDE, N.; VARAMBALLY, R.; GHOSH, D.; BARRETTE, T.; PANDEY, A.; CHINNAIYAN, A. M. Oncomine: a cancer microarray database and integrated data-mining platform. **Neoplasia (New York, NY)**, Neoplasia Press, v. 6, n. 1, p. 1, 2004. Citation on page 30.

ROSENBLATT, F. **The perceptron – a perceiving and recognizing automaton (Project Para)**. 1957. Citation on page 27.

SANDER, J. D.; JOUNG, J. K. Crispr-cas systems for editing, regulating and targeting genomes. **Nature biotechnology**, Nature Publishing Group, v. 32, n. 4, p. 347, 2014. Citation on page 36.

SANTAMARÍA, R.; QUINTALES, L.; THERÓN, R. Methods to bicluster validation and comparison in microarray data. In: SPRINGER. **International Conference on Intelligent Data Engineering and Automated Learning**. [S.l.], 2007. p. 780–789. Citations on pages 44, 46, 48, 53, and 62.

SETUBAL, J. C.; MEIDANIS, J. **Introduction to computational molecular biology**. [S.l.]: PWS Pub. Boston, 1997. Citation on page 28.

SHABALIN, A. A.; WEIGMAN, V. J.; PEROU, C. M.; NOBEL, A. B. *et al.* Finding large average submatrices in high dimensional data. **The Annals of Applied Statistics**, Institute of Mathematical Statistics, v. 3, n. 3, p. 985–1012, 2009. Citation on page 55.

SHAH, S. A. *et al.* Crispr/cas and cmr modules, mobility and evolution of adaptive immune systems. **Research in microbiology**, p. 27–38, 01 2011. Citations on pages 91 and 98.

____. Comprehensive search for accessory proteins encoded with archaeal and bacterial type iii crispr-cas gene cassettes reveals 39 new cas gene families. **RNA Biology**, Taylor & Francis, v. 0, n. 0, p. 1–13, 2018. PMID: 29911924. Citations on pages 67, 68, 71, 74, 83, and 87.

SHMAKOV, S.; ABUDAYYEH, O. O.; MAKAROVA, K. S.; WOLF, Y. I.; GOOTENBERG, J. S.; SEMENOVA, E.; MINAKHIN, L.; JOUNG, J.; KONERMANN, S.; SEVERINOV, K. *et al.* Discovery and functional characterization of diverse class 2 crispr-cas systems. **Molecular cell**, Elsevier, v. 60, n. 3, p. 385–397, 2015. Citations on pages 35, 36, 66, 68, and 84.

SHMAKOV, S. *et al.* Diversity and evolution of class 2 crispr–cas systems. **Nature Reviews Microbiology**, 2017. Citations on pages 66, 68, 74, 83, and 84.

SHU, L.; XU, H.; LIU, B. DOC: Deep open classification of text documents. In: **Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing**. Copenhagen, Denmark: Association for Computational Linguistics, 2017. p. 2911–2916. Available: <https://www.aclweb.org/anthology/D17-1314>. Citations on pages 89, 90, and 91.

SINKUNAS, T.; GASIUNAS, G.; FREMAUX, C.; BARRANGOU, R.; HORVATH, P.; SIKSNYS, V. Cas3 is a single-stranded DNA nuclease and ATP-dependent helicase in the CRISPR/-Cas immune system. **EMBO J**, v. 30, n. 7, p. 1335–42, 2011. ISSN 0261-4189. Citation on page 66.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. **Information Processing & Management**, Elsevier, v. 45, n. 4, p. 427–437, 2009. Citations on pages 70 and 95.

SOUTO, M. C. d.; COSTA, I. G.; ARAUJO, D. S. d.; LUDERMIR, T. B.; SCHLIEP, A. Clustering cancer gene expression data: a comparative study. **BMC bioinformatics**, BioMed Central, v. 9, n. 1, p. 497, 2008. Citation on page 30.

SPELLMAN, P. T.; SHERLOCK, G.; ZHANG, M. Q.; IYER, V. R.; ANDERS, K.; EISEN, M. B.; BROWN, P. O.; BOTSTEIN, D.; FUTCHER, B. Comprehensive identification of cell cycle–regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. **Molecular biology of the cell**, Am Soc Cell Biol, v. 9, n. 12, p. 3273–3297, 1998. Citation on page 56.

SUTTLE, C. A. Environmental microbiology: Viral diversity on the global stage. **Nature Microbiology**, p. 16205–16211, 10 2016. Citation on page 86.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to data mining**. [S.l.]: Pearson Education India, 2006. Citation on page 30.

TANAY, A.; SHARAN, R.; SHAMIR, R. Discovering statistically significant biclusters in gene expression data. **Bioinformatics**, Oxford University Press, v. 18, n. suppl_1, p. S136–S144, 2002. Citations on pages 32, 42, 54, and 57.

____. Biclustering algorithms: A survey. **Handbook of computational molecular biology**, Citeseer, v. 9, n. 1-20, p. 122–124, 2005. Citation on page 43.

TANG, H.; KLOPFENSTEIN, D.; PEDERSEN, B.; FLICK, P.; SATO, K.; RAMIREZ, F.; YUNES, J.; MUNGALL, C. Goatools: tools for gene ontology. **Zenodo. doi**, v. 10, 2015. Citation on page 33.

TATUSOV, R. L.; GALPERIN, M. Y.; NATALE, D. A.; KOONIN, E. V. The cog database: a tool for genome-scale analysis of protein functions and evolution. **Nucleic acids research**, Oxford University Press, v. 28, n. 1, p. 33–36, 2000. Citation on page 92.

TENG, L.; CHAN, L. Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data. **Journal of Signal Processing Systems**, Springer, v. 50, n. 3, p. 267–280, 2008. Citations on pages 44, 50, and 53.

TURING, A. M. Computing machinery and intelligence. **Mind**, v. 59, n. 236, p. 433–460, 1950. Citation on page 27.

TURNER, H.; BAILEY, T.; KRZANOWSKI, W. Improved biclustering of microarray data demonstrated through systematic performance tests. **Computational statistics & data analysis**, Elsevier, v. 48, n. 2, p. 235–254, 2005. Citation on page 54.

VAPNIK, V. **The nature of statistical learning theory**. [S.l.]: Springer-Verlag, 1995. Citation on page 70.

VARMA, S.; SIMON, R. Bias in error estimation when using cross-validation for model selection. **BMC bioinformatics**, BioMed Central, v. 7, n. 1, p. 91, 2006. Citation on page 70.

VESTERGAARD, G.; GARRETT, R. A.; SHAH, S. A. Crispr adaptive immune systems of archaea. **RNA biology**, RNA Biol, 11 2014. Citations on pages 79, 87, 91, 98, and 99.

VORONTSOVA, D.; DATSENKO, K. A.; MEDVEDEVA, S.; BONDY-DENOMY, J.; SAVITSKAYA, E. E.; POUGACH, K.; LOGACHEVA, M.; WIEDENHEFT, B.; DAVIDSON, A.; SEVERINOV, K.; SEMENOVA, E. Foreign DNA acquisition by the I-F CRISPR–Cas system requires all components of the interference machinery. **Nucleic Acids Research**, v. 43, n. 22, p. 10848–10860, 11 2015. ISSN 0305-1048. Available: <https://doi.org/10.1093/nar/gkv1261>. Citation on page 99.

WANG, W.; ZHENG, V. W.; YU, H.; MIAO, C. A survey of zero-shot learning: Settings, methods, and applications. **ACM Transactions on Intelligent Systems and Technology (TIST)**, ACM New York, NY, USA, v. 10, n. 2, p. 1–37, 2019. Citation on page 103.

WANG, Z.; GERSTEIN, M.; SNYDER, M. Rna-seq: a revolutionary tool for transcriptomics. **Nature reviews genetics**, Nature Publishing Group, v. 10, n. 1, p. 57–63, 2009. Citation on page 30.

WATERMAN, M. S. **Introduction to computational biology: maps, sequences and genomes**. [S.l.]: CRC Press, 1995. Citation on page 28.

WESTRA, E. R. *et al.* Crispr immunity relies on the consecutive binding and degradation of negatively supercoiled invader dna by cascade and cas3. **Mol Cell**, p. 595–605, 04 2013. Citation on page 99.

WILLMOTT, C. J.; MATSUURA, K. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. **Climate research**, v. 30, n. 1, p. 79–82, 2005. Citation on page 71.

WU, X.; KUMAR, V.; QUINLAN, J. R.; GHOSH, J.; YANG, Q.; MOTODA, H.; MCLACHLAN, G. J.; NG, A.; LIU, B.; YU, P. S.; ZHOU, Z.-H.; STEINBACH, M.; HAND, D. J.; STEINBERG, D. Top 10 algorithms in data mining. **Knowledge and information systems**, Springer, v. 14, n. 1, p. 1–37, 2008. Citation on page 70.

WU, Y.; LIANG, D.; WANG, Y.; BAI, M.; TANG, W.; BAO, S.; YAN, Z.; LI, D.; LI, J. Correction of a genetic disease in mouse via use of crispr-cas9. **Cell stem cell**, Elsevier, v. 13, n. 6, p. 659–662, 2013. Citation on page 36.

XIAO, N.; CAO, D.-S.; ZHU, M.-F.; XU, Q.-S. protr/protrweb: R package and web server for generating various numerical representation schemes of protein sequences. **Bioinformatics**, Oxford University Press, v. 31, n. 11, p. 1857–1859, 2015. Citation on page 103.

XIE, J.; MA, A.; FENNELL, A.; MA, Q.; ZHAO, J. It is time to apply biclustering: a comprehensive review of biclustering applications in biological and biomedical data. **Briefings in bioinformatics**, Oxford University Press, v. 20, n. 4, p. 1450–1465, 2019. Citation on page 31.

XU, R.; WUNSCH, D. Survey of clustering algorithms. **IEEE Transactions on neural networks**, Ieee, v. 16, n. 3, p. 645–678, 2005. Citation on page 30.

YANG, J.; WANG, W.; WANG, H.; YU, P. δ-clusters: capturing subspace correlation in a large data set. In: IEEE. **IEEE ICDE**. [S.l.], 2002. p. 517–528. Citations on pages 47 and 53.

YANG, W.-H.; DAI, D.-Q.; YAN, H. Finding correlated biclusters from gene expression data. **IEEE TKDE**, IEEE, v. 23, n. 4, p. 568–584, 2011. Citations on pages 50 and 53.

YIP, K. Y.; CHEUNG, D. W.; NG, M. K. Harp: A practical projected clustering algorithm. **IEEE TKDE**, IEEE, v. 16, n. 11, p. 1387–1397, 2004. Citations on pages 48 and 53.

ZHAN, T.; RINDTORFF, N.; BETGE, J.; EBERT, M. P.; BOUTROS, M. Crispr/cas9 for cancer research and therapy. In: ELSEVIER. **Seminars in cancer biology**. [S.l.], 2019. v. 55, p. 106–119. Citation on page 36.

ZHANG, A. **Advanced analysis of gene expression microarray data**. [S.l.]: World Scientific Publishing Company, 2006. Citation on page 30.

ZHANG, J. *et al.* Structure and mechanism of the CMR complex for CRISPR-mediated antiviral immunity. **Mol Cell**, v. 45, n. 3, p. 303–13, 2012. ISSN 1097-2765. Citation on page 67.

ZHANG, Q.; YE, Y. Not all predicted crispr-cas systems are equal: isolated cas genes and classes of crispr like elements. **BMC Bioinformatics**, Public Library of Science, 2 2017. Citations on pages 67, 82, and 87.

APPENDIX

A

SUPPLEMENTARY MATERIAL FOR "CRISPRCASIDENTIFIER: MACHINE LEARNING FOR ACCURATE IDENTIFICATION AND CLASSIFICATION OF CRISPR-CAS SYSTEMS"

This supplementary material is also available online: <https://doi.org/10.1093/gigascience/giaa062>.

In Table 11, we summarize the five different collections of HMM models, labeled HMM_1 ... HMM_5, by listing the number of models for each Cas protein family.

In Table 12, we summarize the percentage of cassettes that are complete for each subtype, ignoring Cas proteins that are contained in less than 5% of the cassettes of each subtype. We observed in the experimental results that, even though some incomplete cassettes are present, the three classifiers were still able to capture the relations among the remaining proteins. The complete results for all sets of HMM models and the two evaluation measures (adjusted balanced accuracy and F-score) are presented in the Supplementary Figure 21.

Concerning our *one-vs-the-rest* experiments we can use, in the case of SVM, the margin separating positive and negative data as an additional quality criteria. In Figure 22, we present an example with subtype I-D, where a clear separation of SVM scores for the positive (I-D) and negative classes (other subtypes) can be observed.

In Figure 23, we present the full *one-vs-the-rest* CART for the I-D subtype. As one can see, a strong evidence for Cas10 immediately points to a subtype I-D (top node and right branch). Otherwise, if we have middle evidence for Cas10, we need at least weak evidence for Cas3 to determine subtype I-D. Finally, if we have only weak evidence for Cas10, we need at least

weak evidence for Cas3 and also for Cas1 to determine subtype I-D (left branch). However, the classification is not pure anymore.

Since the current classification (MAKAROVA *et al.*, 2015) is based only on the interference module, the adaptation-related Cas proteins (Cas1, Cas2 and Cas4) should not have a high importance for our classification pipeline. Thus, we removed, in another experiment, these proteins and the process proteins (Cas6), and tested the predictive performance of our classification pipeline when removing this information. The obtained results were similar to those discussed in our paper and support our discussion and main conclusions (see Supplementary Figure 24), strengthening the hypothesis that our ML-based approach captured biologically relevant information.

In Figures 25–29 we present the remaining regression results, concerning the other subtypes and datasets. In general, we can observe that the proteins can be well predicted. In Figures 30–34, we present regression results considering the full datasets (i.e., not separating by subtype).

In Tables 13–28 we present all association rules found for the HMM1 dataset for each subtype separately.

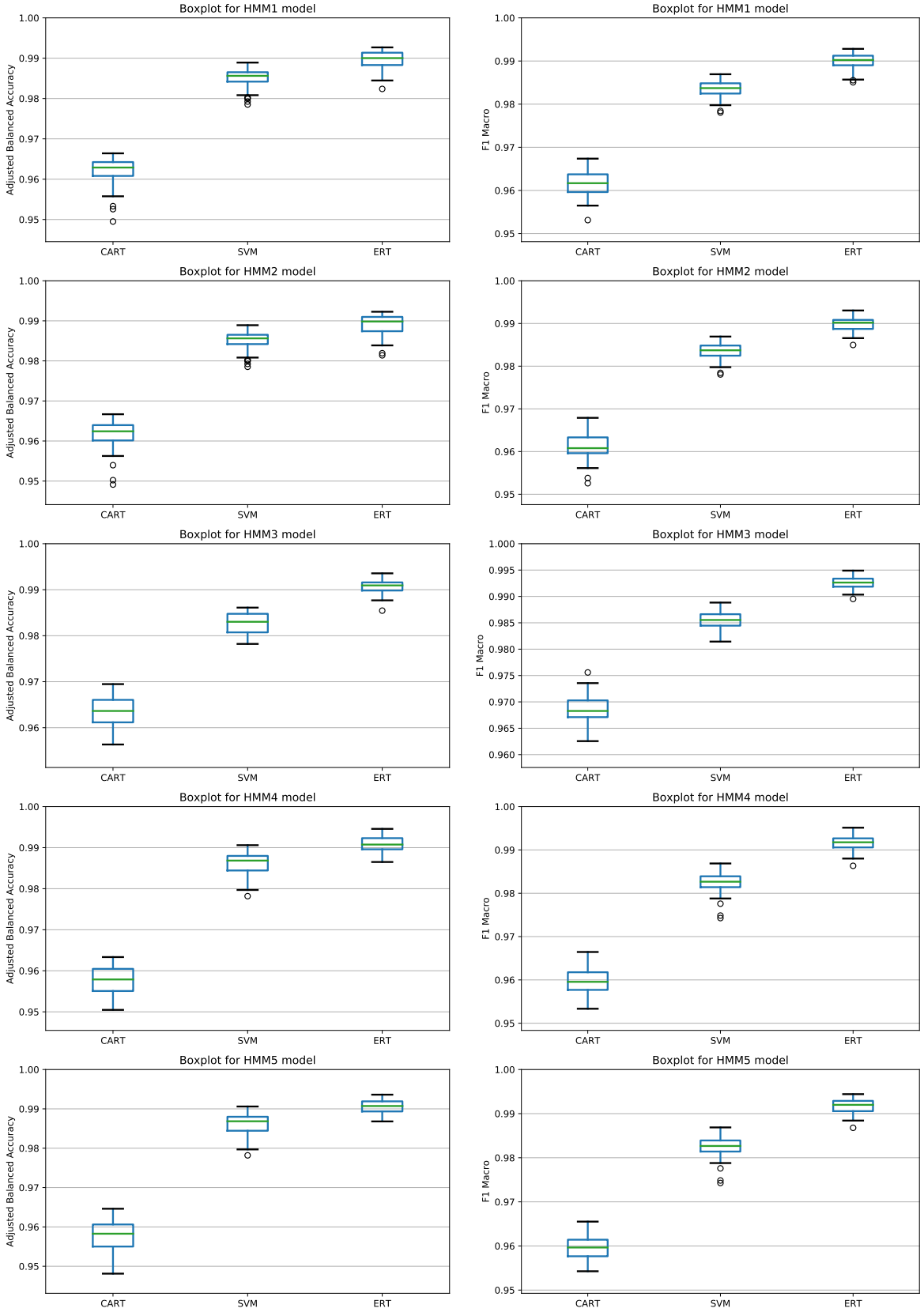
Table 11 – Number of models for each Cas protein family

Cas Protein	# of models HMM1	# of models HMM2	# of models HMM3	# of models HMM4	# of models HMM5
cas1	17	18	15	10	8
cas10	18	18	18	6	6
cas11	2	2	2	2	2
cas12			2	4	4
cas13			3	4	4
cas2	90	93	78	35	38
cas3	25	25	19	14	10
cas4	18	19	19	12	12
cas5	37	35	34	17	17
cas6	36	36	37	4	4
cas7	19	21	19	11	15
cas8	38	38	48	27	20
cas9	4	3	11	6	4
casR	2	1	2	2	2
cmr1	5	5	3	2	2
cmr3	2	2	2	3	3
cmr4	2	2	2	2	2
cmr5	3	3	3	4	4
cmr6	3	3	2	3	3
cmr7	1	1		1	1
cmr8				1	1
cpf1	1	1			
csa3	1	1	1		
csa5	2	2	2	1	1
casX				1	1
csb1	1	1		1	1
csb2	1	1	3	1	1
csb3	1	1		1	1
csc1	1	1	2	1	1
csc2	1	1	1	1	1
cse1				1	1
cse2	2	2	17	1	1
csf1	1	1	1	2	2
csf2	1	1	1	2	2
csf3	1	1	1	2	2
csf4	1	1	1	2	2
csf5			1	1	1
csm2	4	4	4	3	3
csm3	22	25	5	5	5
csm4	3	3	5	2	2
csm5	2	2	2	2	2
csm6	6	6	7	1	1
csn2	5	5	4	1	1
csx			39		
csx10				2	2
csx17				1	1
csx19				1	1
csy1				1	1
csy2				1	1
csy3				1	1
total	379	385	416	209	201

Table 12 – Percentage of complete cassettes across the different subtypes after ignoring Cas proteins that are present in less than 5% of the cassettes of each subtype.

Subtype	% Complete cassettes
I-A	26.72
I-B	71.89
I-C	77.42
I-D	8.7
I-E	80.34
I-F	84.46
I-U	0.0
II-A	93.44
II-B	60.71
II-C	83.79
III-A	29.79
III-B	3.08
III-C	11.83
III-D	1.63
IV-A	0.0
V-A	55.56
VI-B	100.0

Figure 21 – Adjusted balanced accuracy and F-score values achieved for 50 nested ten-fold cross-validation repetitions in all datasets.



(a) Adjusted balanced accuracy.

(b) F-score.

Figure 22 – *One-vs-the-rest* SVM histogram of projections (CHERKASSKY; DHAR, 2010) for the I-D subtype. The solid line represents SVM's optimal hyperplane. The dashed lines represent SVM's margins. The *x*-axis corresponds to the distance of a cassette to the decision boundary. The *y*-axis indicates the frequency of cassettes that have different distances to the decision boundary.

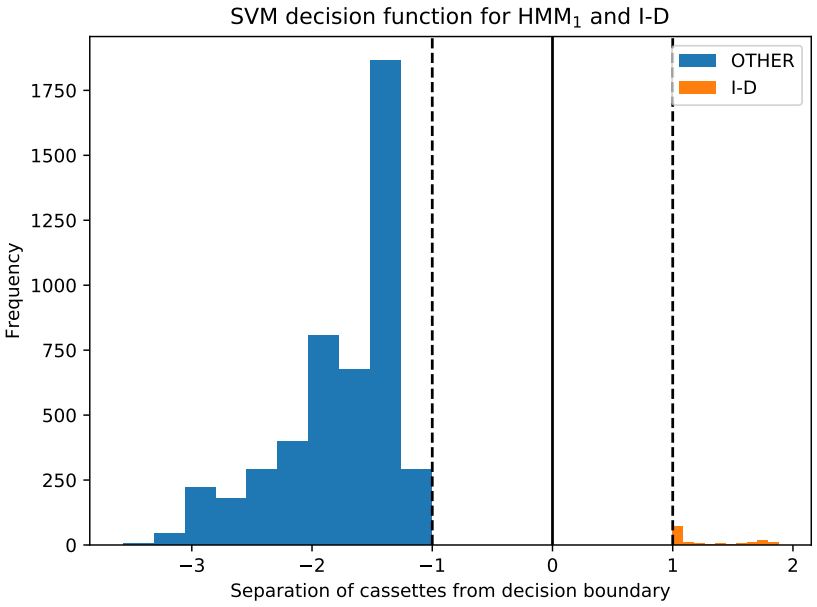


Figure 23 – Full *one-vs-the-rest* CART for the I-D subtype. Cassettes that are labeled as subtype I-D are labeled in blue, the others in brown. Each node shows the fractions of class I-D and other cassettes, indicating the purity of the node. The number of cassettes is shown under "samples" entry.

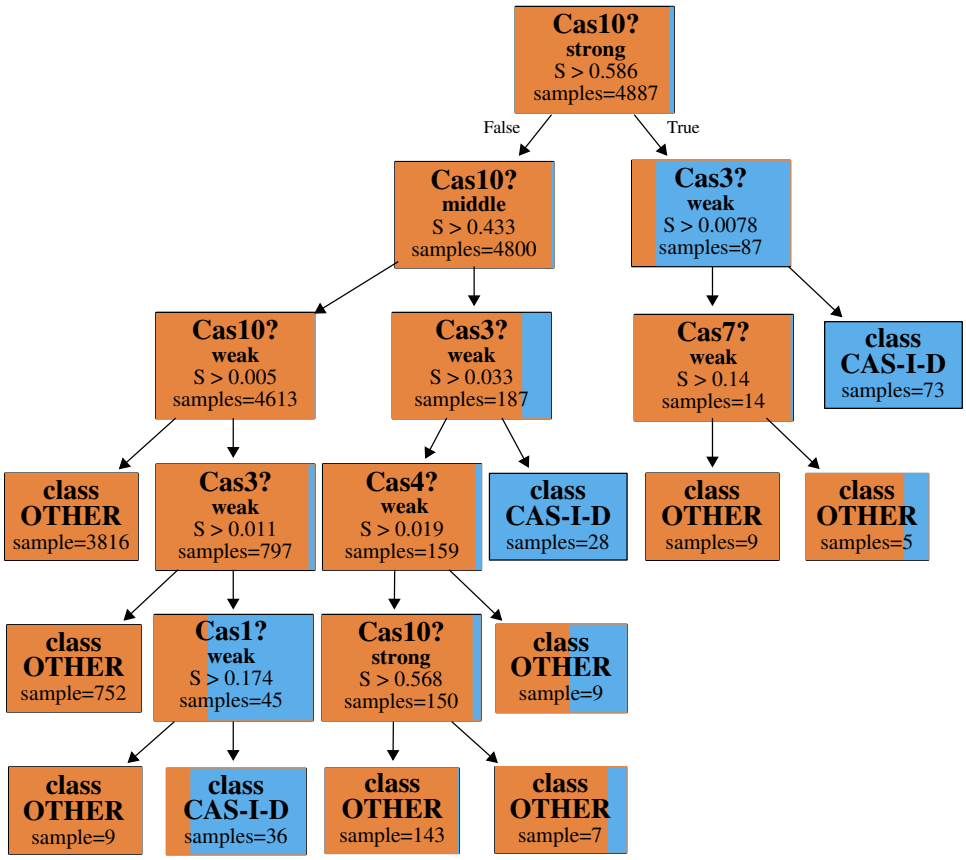
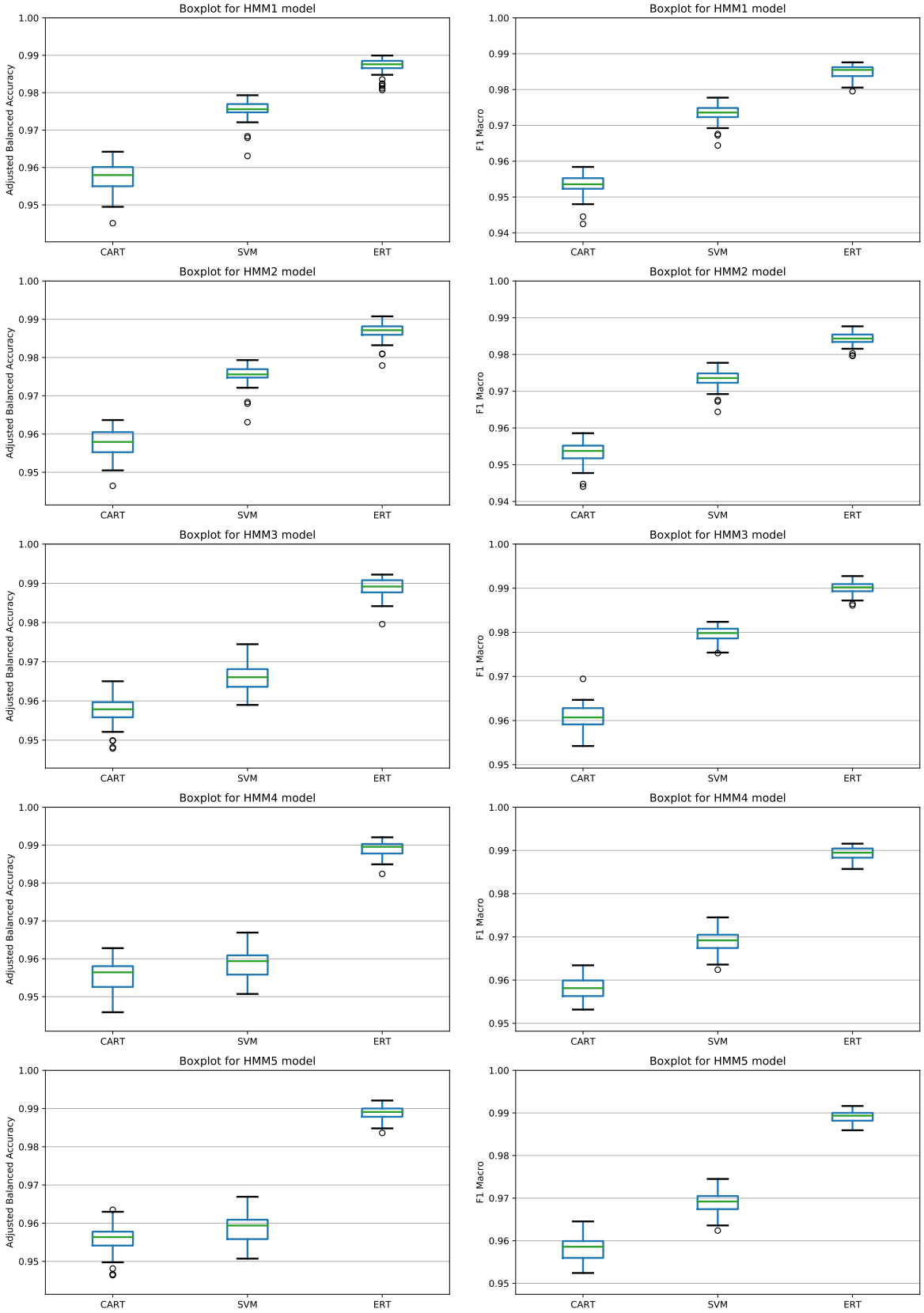


Figure 24 – Adjusted balanced accuracy and F-score values achieved for 50 nested ten-fold cross-validation repetitions in all datasets after removing Cas1, Cas2, Cas4 and Cas6.



(a) Adjusted balanced accuracy.

(b) F-score.

Figure 25 – Mean absolute error results for all subtypes in HMM_1 over 50 nested cross-validation repetitions.

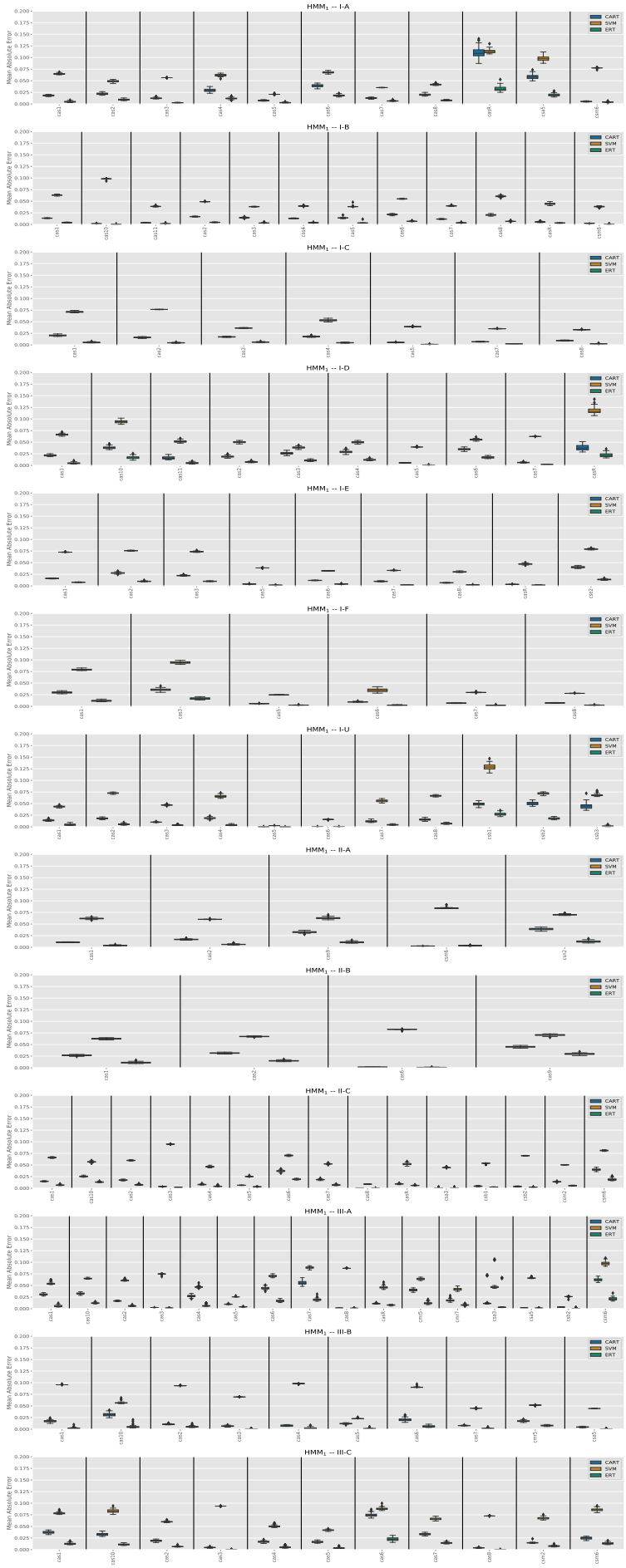


Figure 26 – Mean absolute error results for all subtypes in HMM_2 over 50 nested cross-validation repetitions.

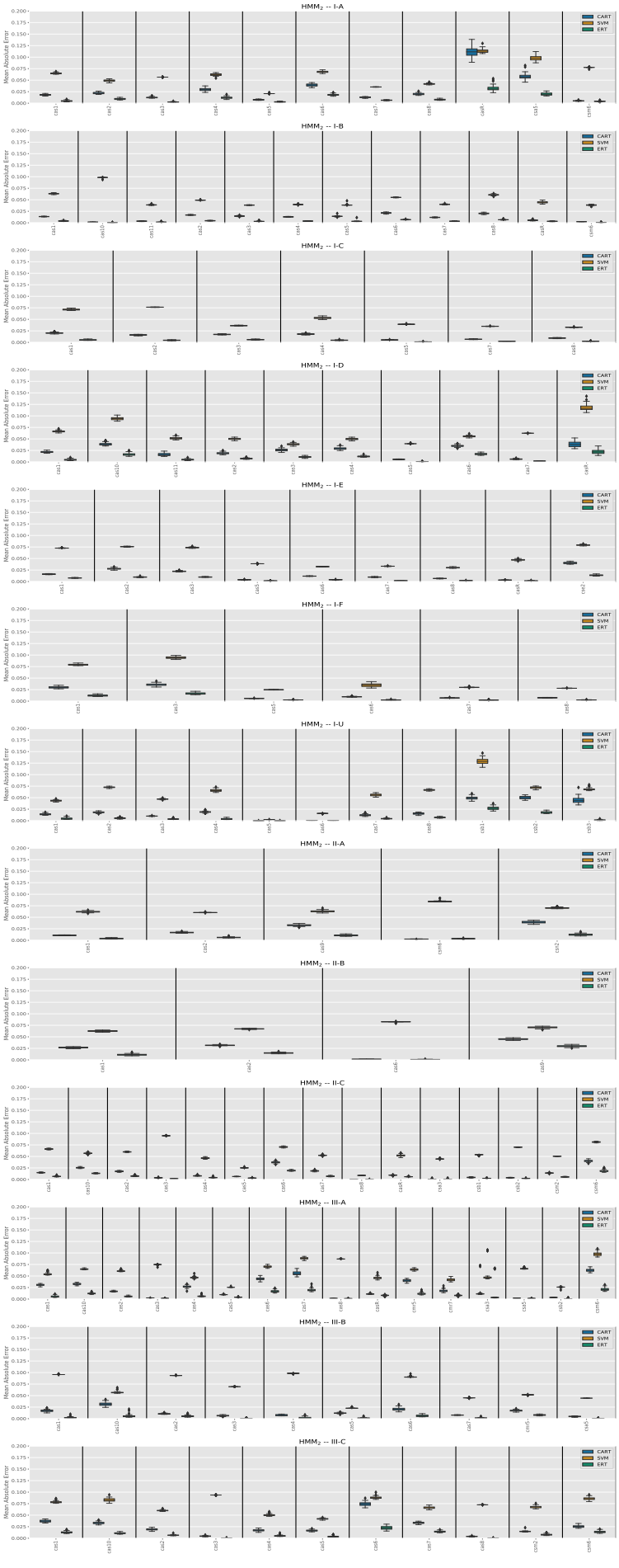


Figure 27 – Mean absolute error results for all subtypes in HMM_3 over 50 nested cross-validation repetitions.

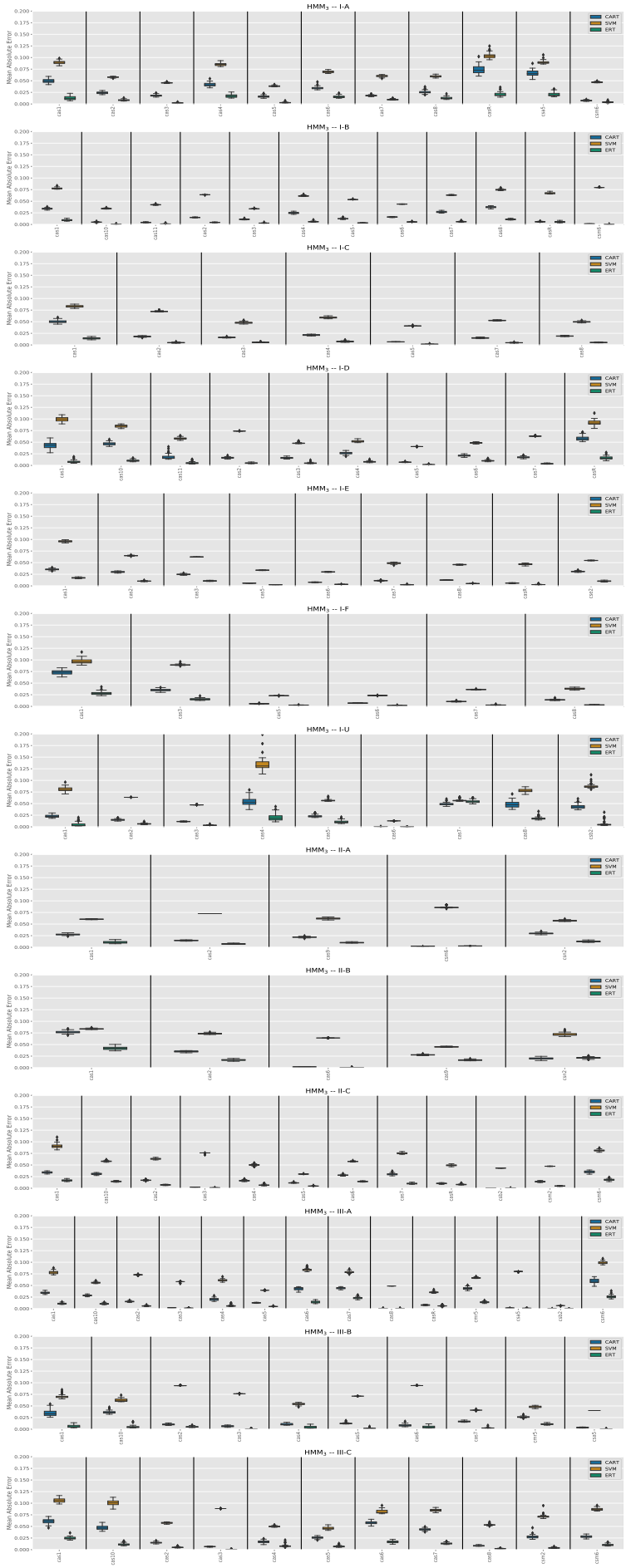


Figure 28 – Mean absolute error results for all subtypes in HMM_4 over 50 nested cross-validation repetitions.

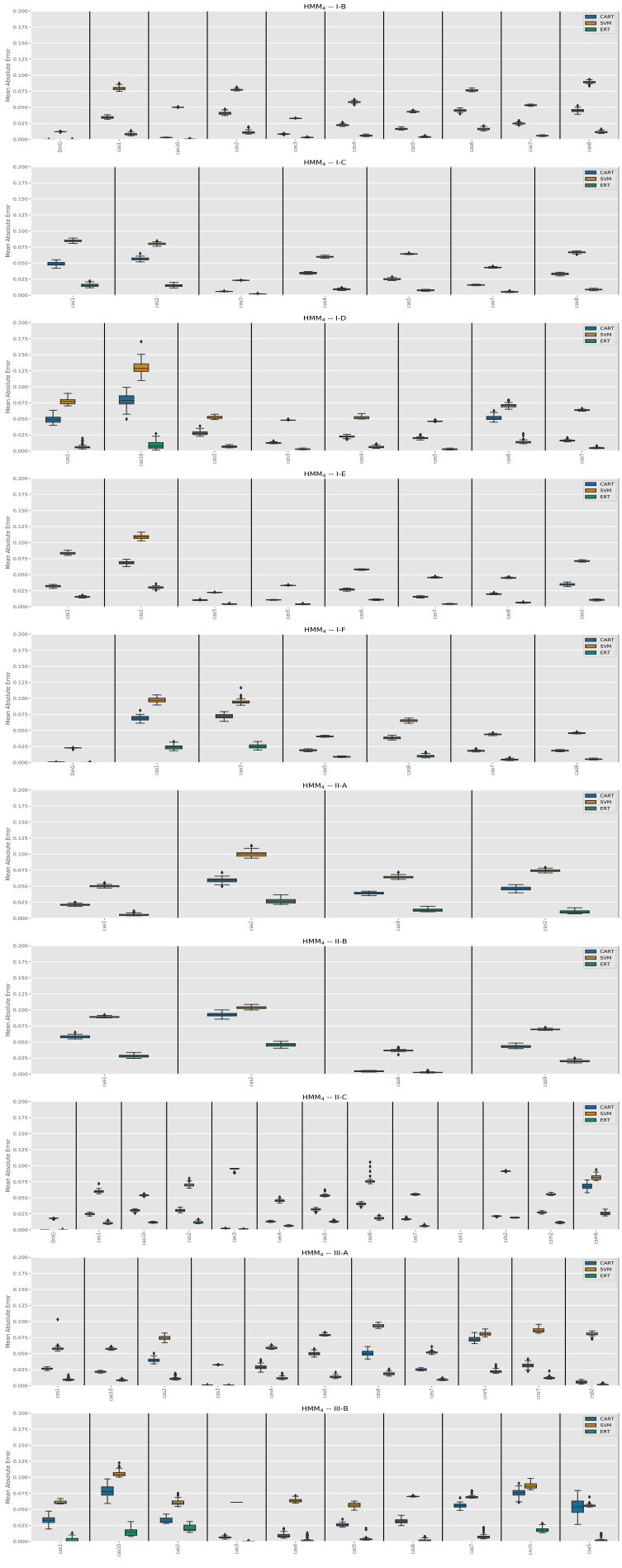
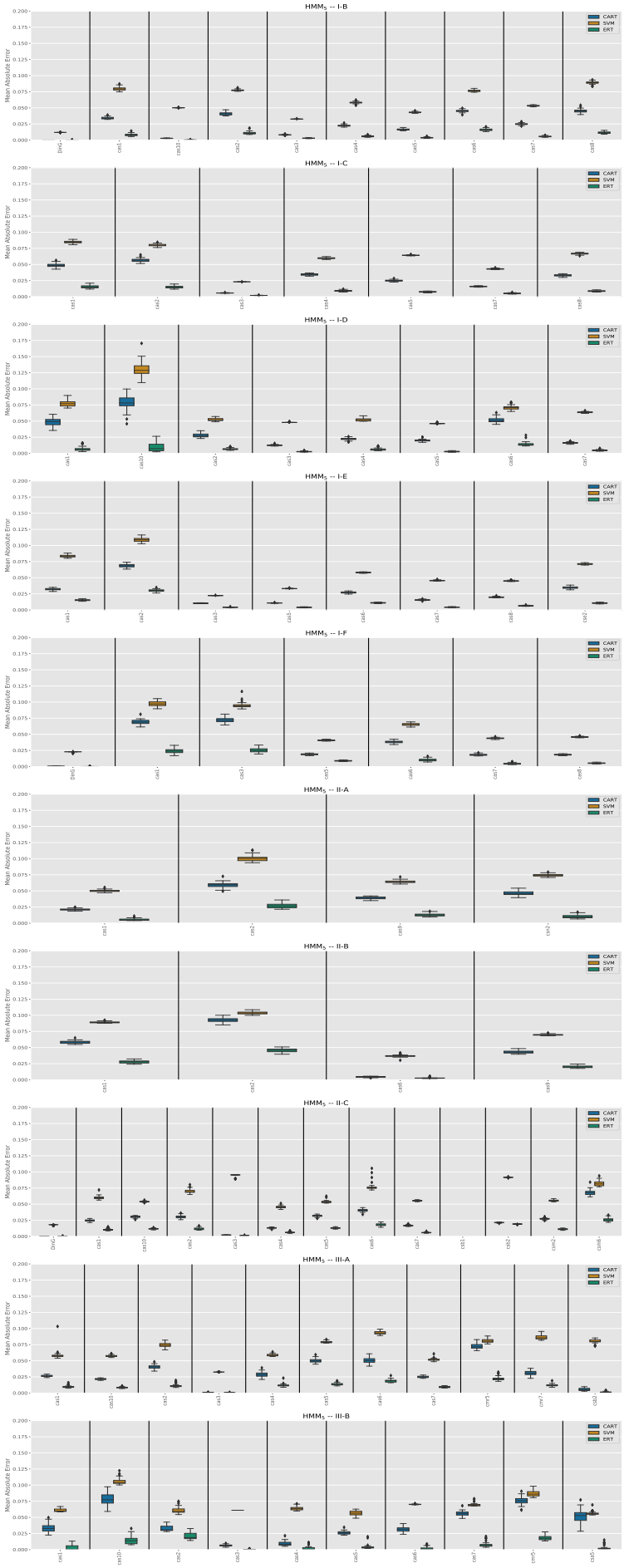


Figure 29 – Mean absolute error results for all subtypes in HMM_5 over 50 nested cross-validation repetitions.



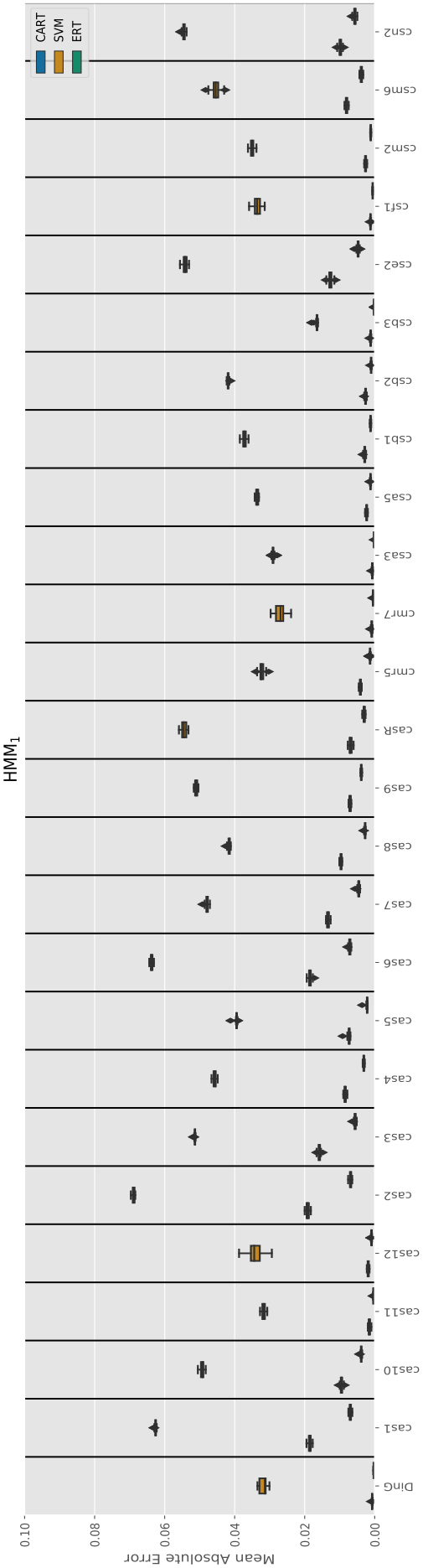
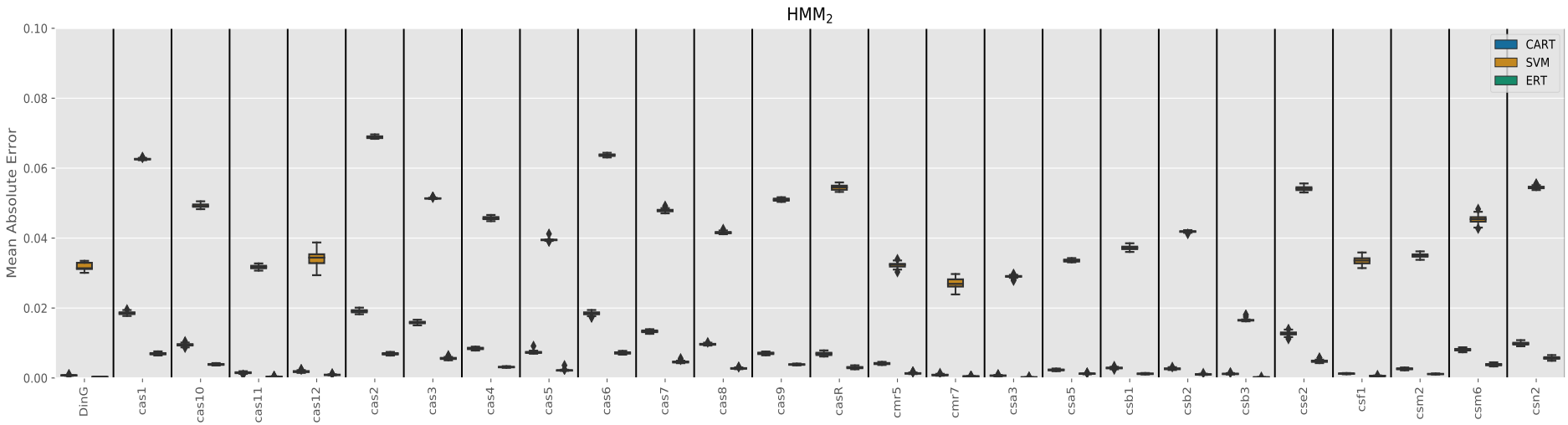


Figure 30 – Mean absolute error results for the full HMM1 dataset (i.e., without separating by subtype) over 50 nested cross-validation repetitions.

Figure 31 – Mean absolute error results for the full HMM2 dataset (i.e., without separating by subtype) over 50 nested cross-validation repetitions.



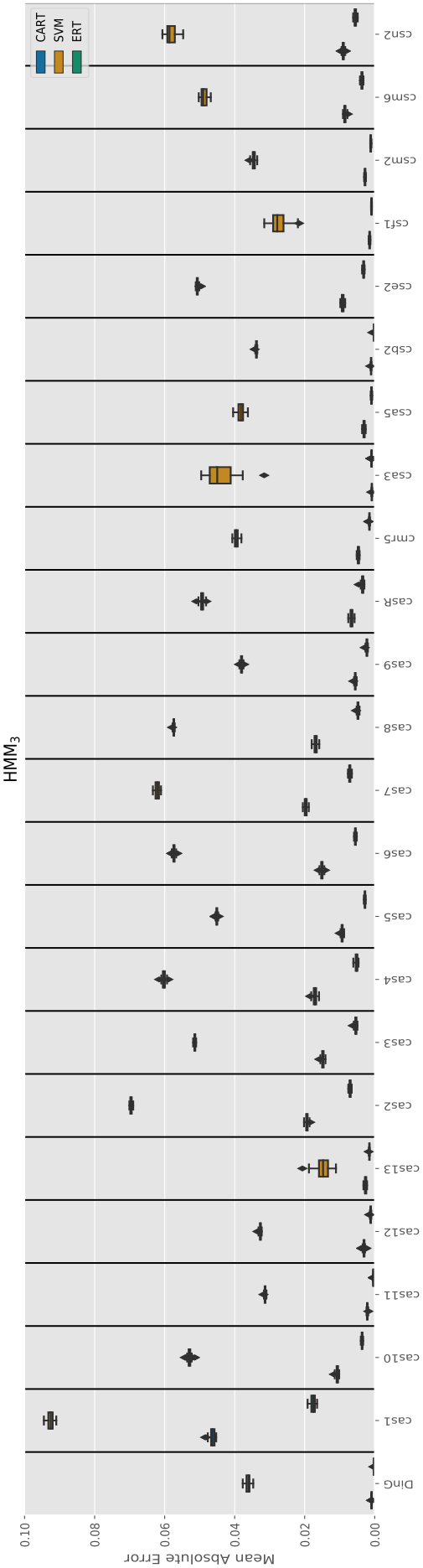
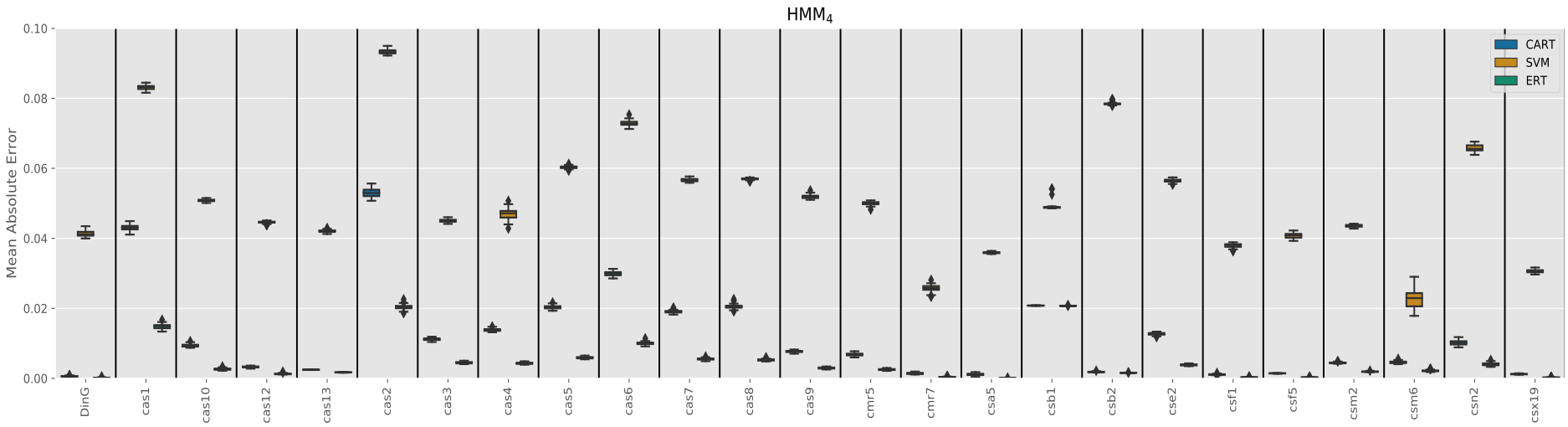


Figure 32 – Mean absolute error results for the full HMM3 dataset (i.e., without separating by subtype) over 50 nested cross-validation repetitions.

Figure 33 – Mean absolute error results for the full HMM4 dataset (i.e., without separating by subtype) over 50 nested cross-validation repetitions.



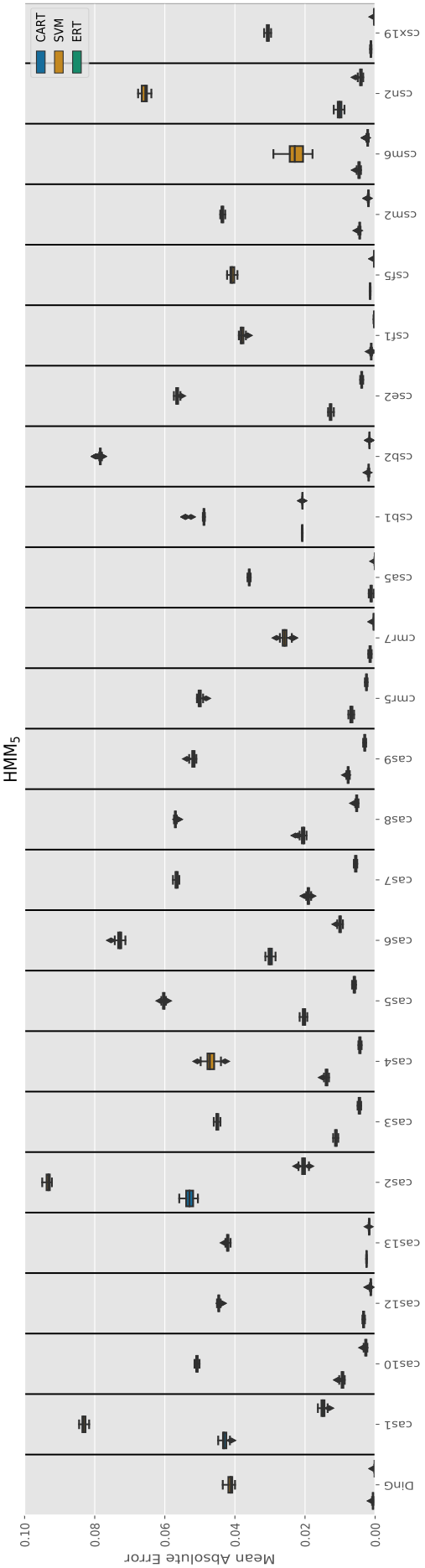


Figure 34 – Mean absolute error results for the full HMM5 dataset (i.e., without separating by subtype) over 50 nested cross-validation repetitions.

Table 13 – Rules generated for subtype I-A.

Target protein	Most important proteins
Cas3	(Cas8, 0.28), (Cas5, 0.24), (Cas7, 0.17), (Csa5, 0.09), (Casr, 0.06), (Cas6, 0.05), (Cas1, 0.04), (Cas4, 0.04), (Cas2, 0.03)
Cas8	(Cas3, 0.21), (Csa5, 0.19), (Casr, 0.18), (Cas5, 0.12), (Cas7, 0.10), (Cas2, 0.06), (Cas4, 0.05), (Cas1, 0.04), (Cas6, 0.04)
Cas7	(Casr, 0.31), (Csa5, 0.21), (Cas8, 0.18), (Cas5, 0.09), (Cas3, 0.07), (Cas1, 0.05), (Cas4, 0.04), (Cas6, 0.03), (Cas2, 0.02)
Cas5	(Casr, 0.30), (Csa5, 0.23), (Cas3, 0.13), (Cas8, 0.09), (Cas6, 0.09), (Cas7, 0.05), (Cas4, 0.04), (Cas1, 0.03), (Cas2, 0.03)
Cas6	(Cas3, 0.52), (Cas5, 0.17), (Cas2, 0.08), (Cas8, 0.07), (Csa5, 0.05), (Cas4, 0.03), (Cas7, 0.03), (Cas1, 0.03), (Casr, 0.02)
Cas1	(Casr, 0.44), (Csa5, 0.11), (Cas4, 0.10), (Cas7, 0.08), (Cas3, 0.08), (Cas8, 0.07), (Cas6, 0.06), (Cas2, 0.03), (Cas5, 0.03)
Cas2	(Cas4, 0.20), (Csa5, 0.16), (Cas7, 0.12), (Cas6, 0.11), (Casr, 0.10), (Cas8, 0.09), (Cas1, 0.08), (Cas3, 0.07), (Cas5, 0.07)
Cas4	(Cas2, 0.21), (Cas3, 0.18), (Cas1, 0.17), (Casr, 0.09), (Cas5, 0.08), (Cas8, 0.08), (Cas7, 0.07), (Csa5, 0.06), (Cas6, 0.05)
Csa5	(Csm6, 0.37), (Casr, 0.18), (Cas3, 0.10), (Cas7, 0.09), (Cas5, 0.08), (Cas6, 0.06), (Cas8, 0.05), (Cas2, 0.03), (Cas1, 0.02), (Cas4, 0.02)
Casr	(Csa5, 0.28), (Cas5, 0.16), (Cas6, 0.12), (Cas3, 0.12), (Cas8, 0.10), (Cas7, 0.09), (Cas2, 0.05), (Cas4, 0.05), (Cas1, 0.03)

Table 14 – Rules generated for subtype I-B.

Target protein	Most important proteins
Cas3	(Cas5, 0.48), (Cas7, 0.19), (Cas8, 0.18), (Cas2, 0.05), (Cas4, 0.04), (Cas6, 0.03), (Cas1, 0.02)
Cas8	(Cas6, 0.28), (Cas3, 0.18), (Cas5, 0.18), (Cas7, 0.14), (Cas2, 0.08), (Cas1, 0.08), (Cas4, 0.06)
Cas7	(Cas3, 0.28), (Cas5, 0.18), (Cas6, 0.12), (Cas4, 0.12), (Cas8, 0.11), (Cas2, 0.10), (Cas1, 0.09)
Cas5	(Cas3, 0.54), (Cas7, 0.16), (Cas8, 0.11), (Cas6, 0.07), (Cas2, 0.05), (Cas4, 0.04), (Cas1, 0.03)
Cas6	(Cas8, 0.37), (Cas7, 0.18), (Cas3, 0.12), (Cas5, 0.11), (Cas2, 0.10), (Cas4, 0.06), (Cas1, 0.06)
Cas1	(Cas4, 0.87), (Cas7, 0.03), (Cas8, 0.03), (Cas3, 0.02), (Cas6, 0.02), (Cas2, 0.02)
Cas2	(Cas8, 0.19), (Cas5, 0.18), (Cas6, 0.14), (Cas7, 0.13), (Cas3, 0.12), (Cas4, 0.11), (Cas1, 0.11), (Casr, 0.02)
Cas4	(Cas1, 0.87), (Cas7, 0.05), (Cas3, 0.02), (Cas8, 0.02), (Cas6, 0.02)
Cas10	(Cas8, 0.28), (Cas5, 0.21), (Cas6, 0.18), (Cas1, 0.18), (Cas2, 0.15)
Casr	(Cas6, 0.23), (Cas8, 0.21), (Cas3, 0.19), (Cas5, 0.15), (Cas1, 0.08), (Cas7, 0.08), (Cas2, 0.03), (Cas4, 0.03)
Cas11	(Cas7, 0.28), (Cas6, 0.21), (Cas8, 0.19), (Cas5, 0.14), (Cas3, 0.09), (Cas4, 0.07), (Cas2, 0.02)
Csm6	(Cas7, 0.36), (Cas4, 0.35), (Cas3, 0.29)

Table 15 – Rules generated for subtype I-C.

Target protein	Most important proteins
Cas3	(Cas7, 0.30), (Cas5, 0.18), (Cas8, 0.17), (Cas4, 0.12), (Cas2, 0.11), (Cas1, 0.11)
Cas8	(Cas7, 0.45), (Cas5, 0.29), (Cas3, 0.08), (Cas2, 0.07), (Cas4, 0.06), (Cas1, 0.04)
Cas7	(Cas5, 0.38), (Cas8, 0.26), (Cas3, 0.22), (Cas2, 0.05), (Cas1, 0.05), (Cas4, 0.04)
Cas5	(Cas7, 0.40), (Cas8, 0.25), (Cas3, 0.10), (Cas2, 0.10), (Cas4, 0.08), (Cas1, 0.07)
Cas1	(Cas4, 0.26), (Cas8, 0.19), (Cas2, 0.18), (Cas3, 0.17), (Cas7, 0.10), (Cas5, 0.09)
Cas2	(Cas5, 0.20), (Cas4, 0.19), (Cas1, 0.19), (Cas8, 0.16), (Cas3, 0.13), (Cas7, 0.12)
Cas4	(Cas2, 0.24), (Cas5, 0.18), (Cas1, 0.18), (Cas8, 0.16), (Cas7, 0.13), (Cas3, 0.12)

Table 16 – Rules generated for subtype I-D.

Target protein	Most important proteins
Cas3	(Cas11, 0.48), (Cas10, 0.20), (Cas5, 0.11), (Cas7, 0.08), (Cas6, 0.04), (Cas4, 0.03), (Cas2, 0.03), (Cas1, 0.02)
Cas7	(Cas5, 0.56), (Cas3, 0.07), (Cas2, 0.07), (Casr, 0.07), (Cas6, 0.07), (Cas10, 0.06), (Cas1, 0.05), (Cas11, 0.03), (Cas4, 0.03)
Cas5	(Cas7, 0.66), (Cas10, 0.12), (Cas3, 0.07), (Cas6, 0.05), (Cas2, 0.04), (Cas4, 0.02), (Casr, 0.02), (Cas1, 0.02)
Cas6	(Cas3, 0.20), (Cas2, 0.19), (Cas10, 0.17), (Cas4, 0.14), (Cas7, 0.12), (Cas5, 0.10), (Casr, 0.03), (Cas1, 0.02)
Cas1	(Cas7, 0.36), (Cas5, 0.15), (Cas4, 0.15), (Cas10, 0.11), (Cas2, 0.08), (Cas3, 0.07), (Casr, 0.05), (Cas6, 0.04)
Cas2	(Cas10, 0.21), (Cas5, 0.20), (Cas3, 0.15), (Cas4, 0.14), (Cas6, 0.12), (Cas7, 0.08), (Casr, 0.05), (Cas1, 0.04)
Cas4	(Cas10, 0.24), (Cas3, 0.18), (Cas2, 0.11), (Cas7, 0.10), (Cas1, 0.10), (Casr, 0.09), (Cas5, 0.08), (Cas6, 0.06), (Cas11, 0.03)
Cas10	(Cas3, 0.28), (Cas5, 0.26), (Cas7, 0.17), (Cas4, 0.08), (Cas2, 0.06), (Cas6, 0.05), (Casr, 0.05), (Cas1, 0.04), (Cas11, 0.02)
Casr	(Cas5, 0.21), (Cas10, 0.18), (Cas2, 0.16), (Cas7, 0.14), (Cas1, 0.12), (Cas4, 0.09), (Cas3, 0.07), (Cas6, 0.04)
Cas11	(Cas3, 0.47), (Cas10, 0.36), (Cas5, 0.15)

Table 17 – Rules generated for subtype I-E.

Target protein	Most important proteins
Cas3	(Cas8, 0.74), (Cse2, 0.08), (Cas7, 0.07), (Cas5, 0.04), (Cas1, 0.03), (Cas6, 0.02)
Cas8	(Cas3, 0.68), (Cse2, 0.13), (Cas5, 0.07), (Cas7, 0.05), (Cas1, 0.03), (Cas6, 0.03), (Cas2, 0.02)
Cse2	(Cas7, 0.25), (Cas5, 0.23), (Cas8, 0.19), (Cas1, 0.13), (Cas3, 0.10), (Cas6, 0.08), (Cas2, 0.02)
Cas7	(Cas6, 0.21), (Cas8, 0.18), (Cse2, 0.16), (Cas3, 0.14), (Cas1, 0.13), (Cas5, 0.11), (Cas2, 0.06)
Cas5	(Cse2, 0.19), (Cas1, 0.19), (Cas6, 0.18), (Cas7, 0.14), (Cas8, 0.13), (Cas2, 0.09), (Cas3, 0.08)
Cas6	(Cas5, 0.26), (Cas2, 0.16), (Cas7, 0.16), (Cas1, 0.15), (Cse2, 0.12), (Cas8, 0.10), (Cas3, 0.05)
Cas1	(Cas7, 0.21), (Cas5, 0.20), (Cse2, 0.18), (Cas3, 0.13), (Cas8, 0.13), (Cas6, 0.10), (Cas2, 0.06)
Cas2	(Cas6, 0.24), (Cas5, 0.19), (Cas7, 0.14), (Cas8, 0.12), (Cas1, 0.12), (Cse2, 0.11), (Cas3, 0.08)
Casr	(Cas1, 0.33), (Cas7, 0.25), (Cas8, 0.21), (Cas5, 0.06), (Cas2, 0.04), (Cas6, 0.04), (Cse2, 0.03), (Cas3, 0.03)

Table 18 – Rules generated for subtype I-F.

Target protein	Most important proteins
Cas3	(Cas8, 0.39), (Cas5, 0.29), (Cas7, 0.15), (Cas1, 0.10), (Cas6, 0.07)
Cas8	(Cas5, 0.67), (Cas3, 0.10), (Cas7, 0.09), (Cas6, 0.07), (Cas1, 0.07)
Cas7	(Cas8, 0.28), (Cas5, 0.20), (Cas6, 0.19), (Cas1, 0.18), (Cas3, 0.16)
Cas5	(Cas8, 0.65), (Cas6, 0.13), (Cas7, 0.11), (Cas1, 0.07), (Cas3, 0.04)
Cas6	(Cas7, 0.28), (Cas8, 0.24), (Cas5, 0.23), (Cas1, 0.20), (Cas3, 0.06)
Cas1	(Cas7, 0.36), (Cas8, 0.19), (Cas5, 0.18), (Cas3, 0.14), (Cas6, 0.13)

Table 19 – Rules generated for subtype I-U.

Target protein	Most important proteins
Cas3	(Csb2, 0.53), (Cas8, 0.33), (Csb1, 0.05), (Cas7, 0.03), (Cas5, 0.02)
Cas8	(Cas3, 0.53), (Csb1, 0.18), (Csb2, 0.15), (Cas7, 0.06), (Cas2, 0.04), (Cas4, 0.03), (Cas1, 0.02)
Cas7	(Csb2, 0.43), (Cas8, 0.29), (Cas3, 0.24), (Cas2, 0.03)
Cas6	(Cas8, 0.28), (Cas3, 0.28), (Csb2, 0.24), (Csb1, 0.20)
Cas1	(Cas4, 0.85), (Cas3, 0.06), (Csb2, 0.04), (Cas8, 0.02), (Cas2, 0.02)
Cas2	(Csb3, 0.34), (Cas1, 0.22), (Cas4, 0.19), (Cas8, 0.14), (Cas3, 0.05), (Csb2, 0.04)
Cas4	(Cas1, 0.86), (Cas2, 0.06), (Cas3, 0.03), (Cas8, 0.02), (Csb2, 0.02)
Csb2	(Cas3, 0.72), (Cas8, 0.17), (Csb1, 0.02), (Cas2, 0.02), (Cas1, 0.02), (Cas4, 0.02)
Csb1	(Cas8, 0.62), (Cas3, 0.11), (Cas2, 0.11), (Csb3, 0.07), (Csb2, 0.06), (Cas1, 0.02)
Csb3	(Cas2, 0.33), (Cas1, 0.17), (Csb1, 0.17), (Cas4, 0.12), (Csb2, 0.12), (Cas3, 0.08)

Table 20 – Rules generated for subtype II-A.

Target protein	Most important proteins
Cas1	(Csn2, 0.42), (Cas9, 0.36), (Cas2, 0.23)
Cas2	(Cas9, 0.55), (Csn2, 0.27), (Cas1, 0.18)
Cas9	(Cas1, 0.41), (Cas2, 0.32), (Csn2, 0.27)
Csn2	(Cas1, 0.62), (Cas9, 0.23), (Cas2, 0.15)

Table 21 – Rules generated for subtype II-B.

Target protein	Most important proteins
Cas1	(Cas9, 0.88), (Cas4, 0.07), (Cas2, 0.05)
Cas2	(Cas9, 0.46), (Cas4, 0.41), (Cas1, 0.14)
Cas4	(Cas9, 0.83), (Cas1, 0.09), (Cas2, 0.08)
Cas9	(Cas1, 0.66), (Cas4, 0.32), (Cas2, 0.02)

Table 22 – Rules generated for subtype II-C.

Target protein	Most important proteins
Cas6	(Cas2, 0.37), (Cas1, 0.36), (Cas9, 0.27)
Cas1	(Cas9, 0.75), (Cas2, 0.25)
Cas2	(Cas1, 0.59), (Cas9, 0.41)
Cas9	(Cas1, 0.68), (Cas2, 0.32)

Table 23 – Rules generated for subtype III-A.

Target protein	Most important proteins
Cas3	(Csb2, 0.23), (Csb1, 0.22), (Cas7, 0.21), (Cas4, 0.13), (Cas2, 0.07), (Cas6, 0.06), (Cas1, 0.04), (Casr, 0.02), (Cas8, 0.02)
Cas7	(Cas5, 0.38), (Cas10, 0.19), (Cas6, 0.13), (Csm2, 0.11), (Cas2, 0.08), (Cas1, 0.05), (Csm6, 0.03)
Cas5	(Cas6, 0.27), (Cas10, 0.25), (Cas7, 0.16), (Csm2, 0.12), (Cas2, 0.08), (Cas1, 0.06), (Csm6, 0.06)
Cas6	(Cas10, 0.27), (Csm6, 0.25), (Cas5, 0.17), (Csm2, 0.13), (Cas2, 0.12), (Cas7, 0.04), (Cas1, 0.02)
Cas1	(Cas4, 0.25), (Cas2, 0.19), (Cas5, 0.13), (Csm2, 0.11), (Cas10, 0.08), (Cas7, 0.08), (Cas6, 0.08), (Csm6, 0.04), (Casr, 0.02)
Cas2	(Csm6, 0.38), (Cas6, 0.21), (Cas10, 0.11), (Cas7, 0.09), (Csm2, 0.08), (Cas1, 0.07), (Cas5, 0.05)
Cas4	(Cas1, 0.73), (Cas7, 0.08), (Cas6, 0.05), (Cas5, 0.04), (Cas2, 0.04), (Cas10, 0.02), (Csm2, 0.02)
Cas10	(Cas5, 0.40), (Cas6, 0.24), (Csm2, 0.10), (Cas7, 0.09), (Csm6, 0.07), (Cas2, 0.04), (Cas1, 0.03)
Casr	(Cas2, 0.27), (Cas3, 0.22), (Cas8, 0.19), (Cas6, 0.10), (Cas1, 0.07), (Cas10, 0.07), (Cas4, 0.06), (Cas5, 0.02), (Csm2, 0.02)
Csm6	(Cas6, 0.48), (Cas7, 0.19), (Cas2, 0.15), (Cas10, 0.08), (Csm2, 0.05), (Cas5, 0.03)
Csm2	(Cas5, 0.27), (Cas6, 0.22), (Cas10, 0.16), (Cas7, 0.12), (Cas2, 0.07), (Csm6, 0.06), (Cas1, 0.05)

Table 24 – Rules generated for subtype III-B.

Target protein	Most important proteins
Cas3	(Casr, 0.19), (Cas7, 0.16), (Cmr5, 0.16), (Csm6, 0.15), (Cas1, 0.15), (Cas4, 0.12), (Cas2, 0.07)
Cas7	(Cas10, 0.24), (Cas2, 0.21), (Cas5, 0.19), (Cmr5, 0.12), (Csm6, 0.09), (Cas6, 0.08), (Cas1, 0.06)
Cas5	(Cmr5, 0.32), (Cas10, 0.24), (Cas7, 0.21), (Cas2, 0.07), (Cas6, 0.05), (Cas1, 0.04), (Csm6, 0.03), (Cas4, 0.02)
Cas6	(Cas1, 0.79), (Cas2, 0.07), (Cmr5, 0.05), (Cas5, 0.04), (Cas10, 0.02), (Cas7, 0.02)
Cas1	(Cas6, 0.84), (Csb2, 0.04), (Cas4, 0.04), (Cas2, 0.04)
Cas2	(Cas7, 0.20), (Cas5, 0.18), (Cas6, 0.16), (Cas10, 0.13), (Cas1, 0.11), (Cmr5, 0.10), (Csm6, 0.07), (Cas4, 0.03), (Csa3, 0.02)
Cas4	(Csb2, 0.67), (Cas1, 0.22), (Cmr5, 0.03), (Cas2, 0.03), (Cas7, 0.02)
Cas10	(Cas7, 0.31), (Cas5, 0.22), (Cas6, 0.16), (Cmr5, 0.11), (Cas1, 0.08), (Cmr7, 0.06), (Cas2, 0.04), (Csm6, 0.03)
Casr	(Cmr5, 0.21), (Cas7, 0.17), (Cas3, 0.16), (Csm6, 0.14), (Cas10, 0.13), (Cas5, 0.07), (Cas1, 0.05), (Cas4, 0.05)
Csm6	(Cmr5, 0.27), (Cas5, 0.26), (Cas10, 0.20), (Cas7, 0.12), (Cas6, 0.08), (Cas1, 0.03), (Cas2, 0.02)
Cmr5	(Cas5, 0.32), (Cas7, 0.21), (Cas10, 0.17), (Cas1, 0.13), (Csm6, 0.10), (Cas2, 0.04), (Cas6, 0.02)
Cmr7	(Cas7, 0.33), (Cmr5, 0.30), (Cas10, 0.21), (Cas5, 0.16)
Csa3	(Cas7, 0.28), (Cas6, 0.16), (Cmr5, 0.12), (Cas4, 0.12), (Cas1, 0.12), (Cas5, 0.12), (Cas2, 0.08)

Table 25 – Rules generated for subtype III-C.

Target protein	Most important proteins
Cas7	(Cas10, 0.46), (Cas5, 0.38), (Cas2, 0.05), (Cmr5, 0.04), (Cas1, 0.04), (Cas6, 0.02)
Cas5	(Cas3, 0.31), (Csa5, 0.27), (Cmr5, 0.16), (Cas7, 0.13), (Cas10, 0.06), (Cas2, 0.03), (Cas6, 0.02)
Cas6	(Cas1, 0.33), (Cas10, 0.31), (Cas7, 0.16), (Cmr5, 0.09), (Cas2, 0.05), (Cas5, 0.04), (Cas4, 0.02)
Cas1	(Cas6, 0.49), (Cas4, 0.20), (Cas2, 0.10), (Cas7, 0.06), (Cas10, 0.06), (Cmr5, 0.06), (Cas5, 0.02)
Cas2	(Cas7, 0.29), (Cas6, 0.23), (Cas10, 0.17), (Cas1, 0.12), (Cas4, 0.09), (Cas5, 0.07), (Cmr5, 0.02)
Cas4	(Cas3, 0.19), (Cas7, 0.16), (Csa5, 0.13), (Cas5, 0.13), (Cas6, 0.12), (Cmr5, 0.10), (Cas1, 0.09), (Cas10, 0.08)
Cas10	(Cas7, 0.46), (Cas5, 0.29), (Cmr5, 0.14), (Cas6, 0.05), (Cas2, 0.03), (Cas1, 0.02)
Cmr5	(Cas5, 0.34), (Cas10, 0.25), (Cas7, 0.17), (Cas2, 0.09), (Cas6, 0.08), (Cas1, 0.06)

Table 26 – Rules generated for subtype III-D.

Target protein	Most important proteins
Cas3	(Cas6, 0.20), (Cas4, 0.13), (Cas2, 0.13), (Cas1, 0.13), (Cas8, 0.13), (Cas10, 0.10), (Cas7, 0.10), (Cas5, 0.08)
Cas8	(Cas3, 0.51), (Cas5, 0.41), (Cas2, 0.06)
Cas7	(Cas10, 0.36), (Csm2, 0.27), (Csm6, 0.15), (Cas5, 0.10), (Cas6, 0.04), (Cas1, 0.03), (Cas2, 0.03)
Cas5	(Cas6, 0.33), (Cas1, 0.25), (Cas2, 0.15), (Cas7, 0.09), (Cas10, 0.06), (Cas3, 0.03), (Cas4, 0.03), (Csm6, 0.02)
Cas6	(Cas10, 0.34), (Cas1, 0.32), (Cas7, 0.09), (Csm6, 0.08), (Csm2, 0.07), (Cas5, 0.06), (Cas2, 0.02)
Cas1	(Cas6, 0.48), (Cas5, 0.26), (Cas4, 0.10), (Cas7, 0.06), (Cas10, 0.05), (Cas2, 0.04)
Cas2	(Cas10, 0.17), (Csm2, 0.15), (Cas7, 0.13), (Cas5, 0.13), (Csm6, 0.10), (Cas8, 0.09), (Cas1, 0.08), (Cas4, 0.07), (Cas6, 0.07), (Cas3, 0.02)
Cas4	(Cas1, 0.48), (Cas6, 0.26), (Cas2, 0.12), (Cas7, 0.07), (Cas5, 0.04)
Cas10	(Csm2, 0.50), (Cas7, 0.21), (Cas5, 0.14), (Cas6, 0.09), (Csm6, 0.04)
Csm6	(Cas5, 0.26), (Cas2, 0.19), (Cas10, 0.18), (Cas1, 0.17), (Cas7, 0.08), (Cas6, 0.07), (Csm2, 0.05)
Csm2	(Cas6, 0.47), (Cas10, 0.33), (Cas7, 0.12), (Cas5, 0.04), (Csm6, 0.03)

Table 27 – Rules generated for subtype IV-A.

Target protein	Most important proteins
Cas7	(Ding, 0.34), (Csf1, 0.28), (Cas5, 0.20), (Cas6, 0.18)
Cas5	(Csf1, 0.46), (Cas7, 0.30), (Ding, 0.19), (Cas6, 0.04)
Cas6	(Cas7, 0.68), (Csf1, 0.20), (Cas5, 0.12)
Ding	(Cas7, 0.64), (Csf1, 0.26), (Cas5, 0.10)
Csf1	(Ding, 0.33), (Cas7, 0.30), (Cas5, 0.25), (Cas6, 0.11)

Table 28 – Rules generated for subtype V-A.

Target protein	Most important proteins
Cas1	(Cas2, 0.64), (Cas4, 0.25), (Cas12, 0.11)
Cas2	(Cas1, 0.54), (Cas4, 0.38), (Cas12, 0.08)
Cas4	(Cas1, 0.63), (Cas2, 0.33), (Cas12, 0.04)
Cas12	(Cas2, 0.50), (Cas1, 0.30), (Cas4, 0.20)

Table 29 – Example for using non-custom HMM models such as PFAM or TIGRFAM

I-A proteins	# of proteins[2]	Our tool	TIGRFAMs	PFam.
cas1	36	36	36	36
cas2	39	39	39	18
cas3	85	85	52	25
cas4	70	70	62	45
cas5	49	49	31	25
cas6	38	38	30	17
cas7	117	117	53	53
cas8	59	55	19	14

Table S19. Summary of the compared tools.

	Identification of Cas proteins	Classification of complete cassettes	Identification of missing Cas proteins	Classification of incomplete cassettes	Learns association rules	Handles unseen cassettes	Method	Input
CRISPRone	Yes	Yes	No	No	No	No	HMM and HMMER tool	DNA only
HmmCas	Yes	No	No	No	No	No	HMM and HMMER tool	Cassette of proteins
CRISPRminer	Yes	Yes	No	No	No	No	HMM and CRISPRcasFinder	DNA only
CRISPRminer2	Yes	Yes	No	No	No	No	HMM, PSI-Blast and CRISPRcasFinder	DNA only
Macsyfinder	Yes	Yes	No	No	No	No	HMM and HMMER tool	Proteins
CRISPRcasFinder	Yes	Yes	No	No	No	No	HMM and Macsyfinder	DNA only
CRISPRdisco	Yes	Yes	No	No	No	No	PSI-Blast	DNA and Proteins
CRISPRcasIdentifier	Yes	Yes	Yes	Yes	Yes	Yes	HMM and 3 different ML approaches	DNA or Proteins

APPENDIX

B

SUPPLEMENTARY MATERIAL FOR "CASBOUNDARY: AUTOMATED DEFINITION OF INTEGRAL CAS CASSETTES"

In Table 30, we show the distribution of CRISPR cassettes into the 22 different subtypes contained in the collected dataset.

In Figures 35 and 36 we present the remaining results for the cassette boundary detection, using a combination of the general HMM features with the protein properties features and using only the protein properties features, respectively.

In Figures 38–46 we present the additional results for Cas type classification.

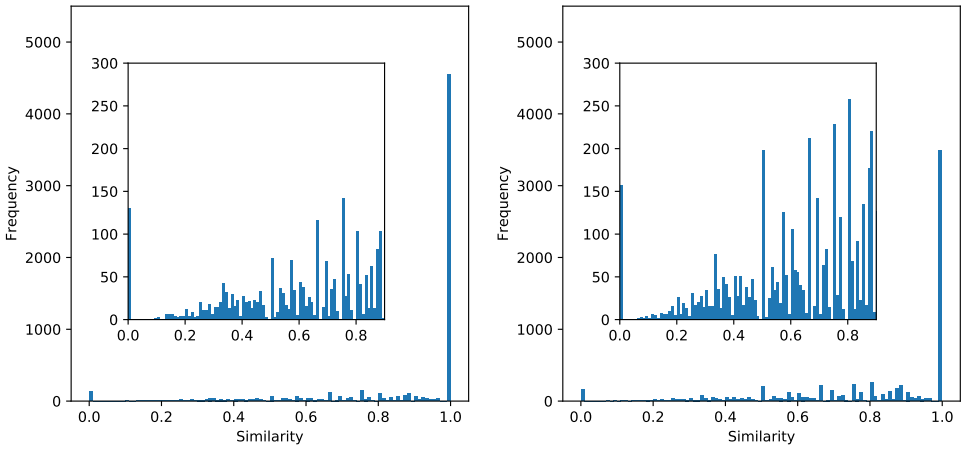
In Figures 47–49 we show the results for the study case on the identification of potentially new Cas proteins.

Finally, in Table 31, we show the results of the occurrence of exchangeable modules.

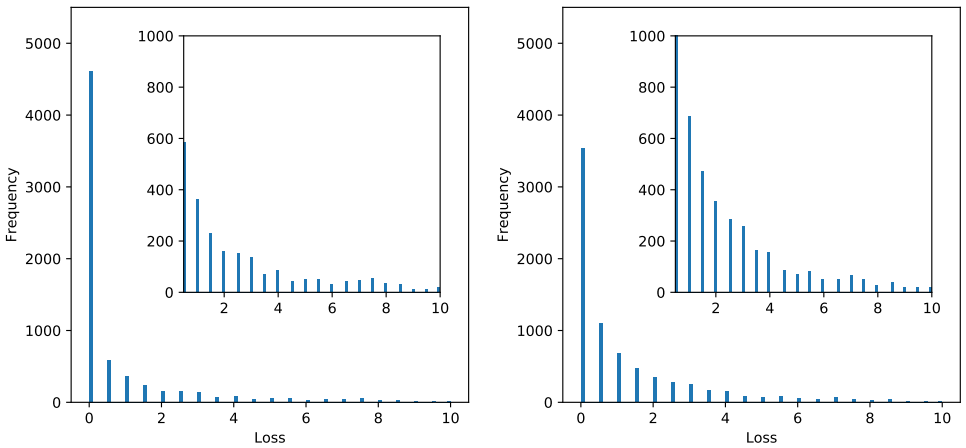
Table 30 – Number of cassettes for each CRISPR subtype in the collected dataset.

CRISPR Subtype	# of cassettes	CRISPR Subtype	# of cassettes
I-A	146	III-B	377
I-B	976	III-C	72
I-C	855	III-D	257
I-D	135	III-E	1
I-E	2122	IV-A	105
I-F	698	V-A	38
I-U	169	V-F	27
II-A	722	VI-A	6
II-B	60	VI-B	53
II-C	563	VI-C	5
III	1	VI-D	1
III-A	518		

Figure 35 – Histogram of the JS and CL for single cassettes using general HMM and protein properties features.



(a) ERT (b) DNN



(c) ERT (d) DNN

Figure 36 – Histogram of the JS and CL for single cassettes using protein properties features.

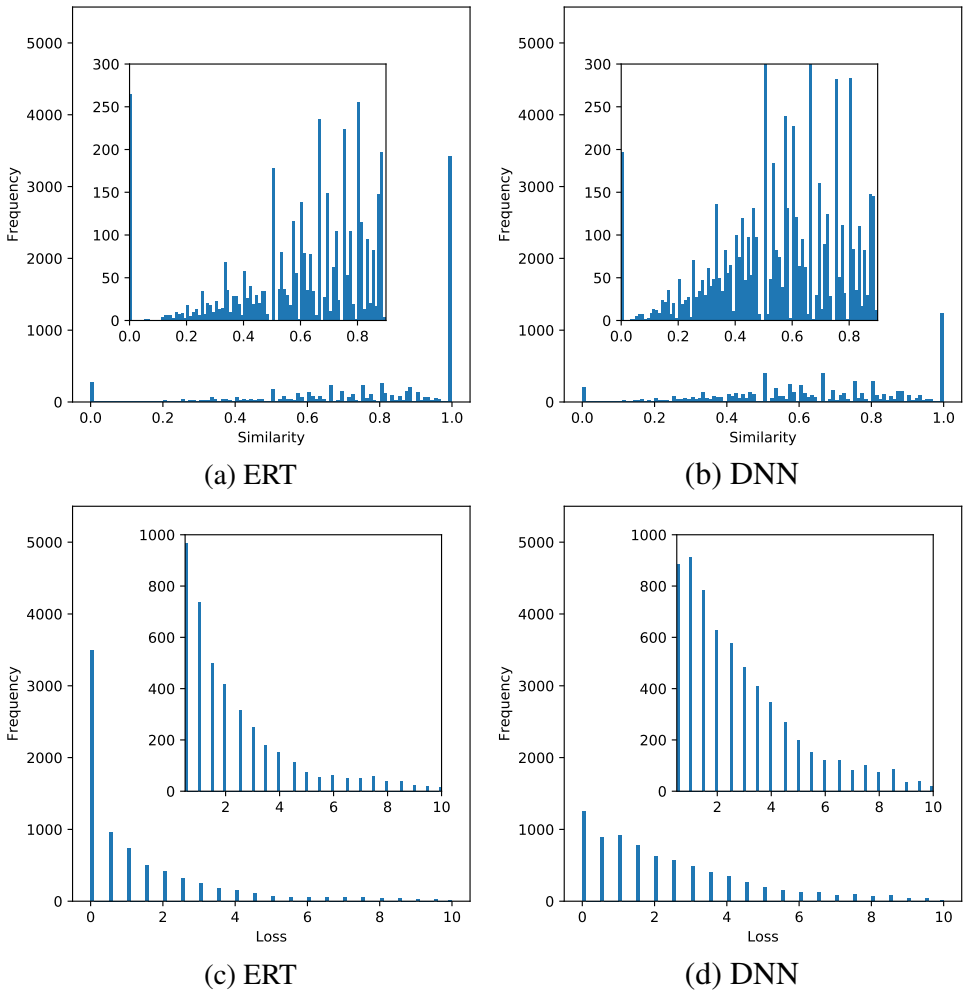


Figure 37 – Comparison between (a) our method's and (b) CRISPRCasFinder's cassette prediction for the organism Thermotoga sp. RQ2. The genome has two single cassettes and one multi-module cassette. Our tool can easily handle those cassettes by identifying them as it should be in nature. In contrast, CRISPRCasFinder struggles to report such cases. For example, in cassette 1 (orange) and cassette 2 (green), one gene is missing. Moreover, cassette 3 and 4 must be one cassette containing three different modules (two interference modules and a single adaptation module). However, CRISPRCasFinder splits it into two different cassettes, which is different to what is expected in nature.

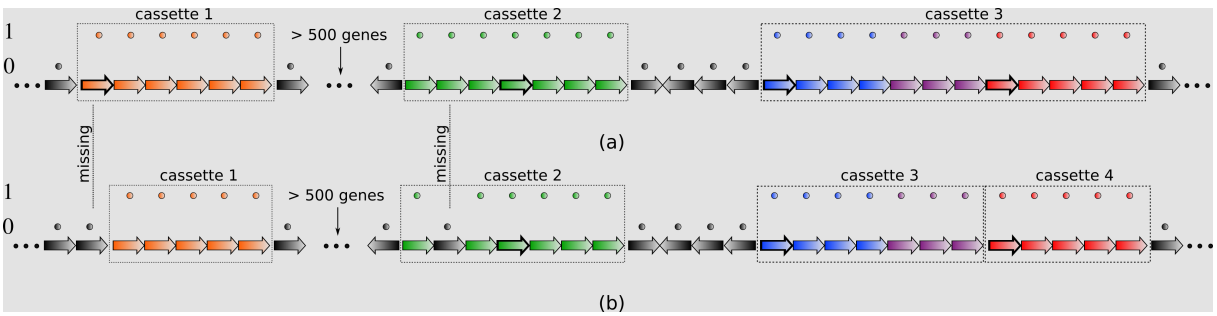


Figure 38 – Cas type prediction F-scores with 3 cas types left out, using a combination of the specific HMM and protein properties features.

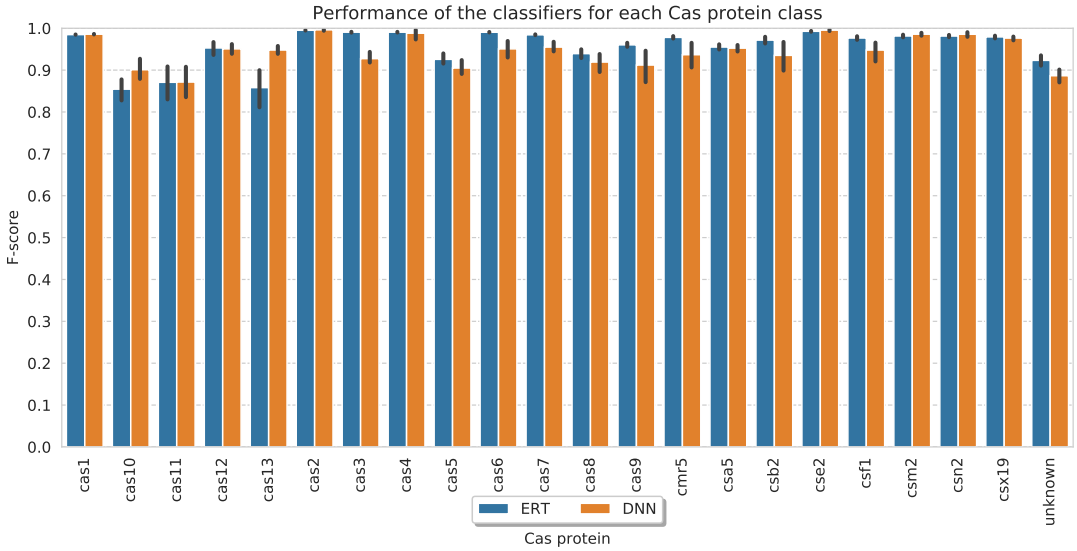


Figure 39 – Cas type prediction F-scores with 1 cas type left out, using the specific HMM features.

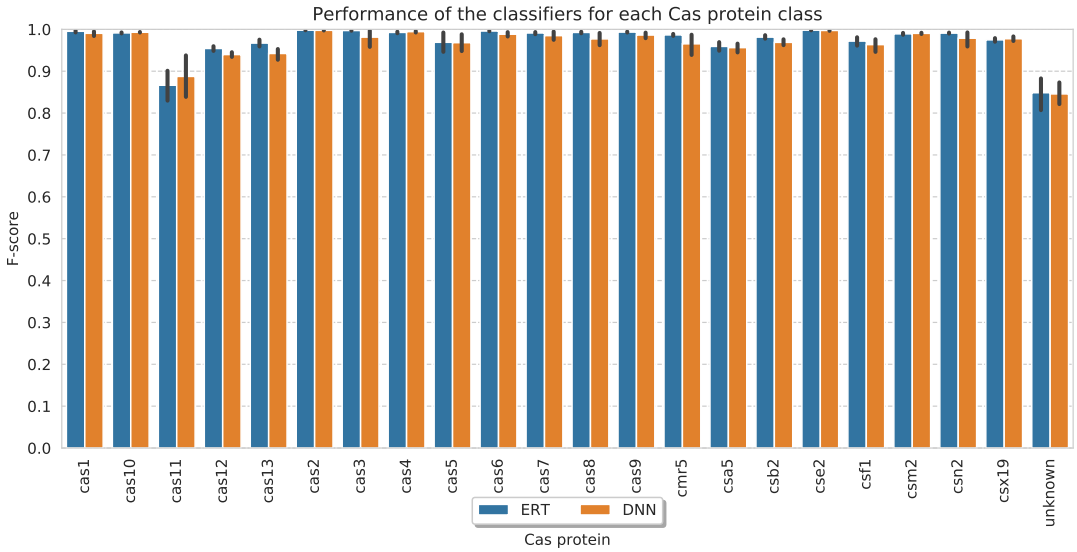


Figure 40 – Cas type prediction F-scores with 3 cas types left out, using the specific HMM features.

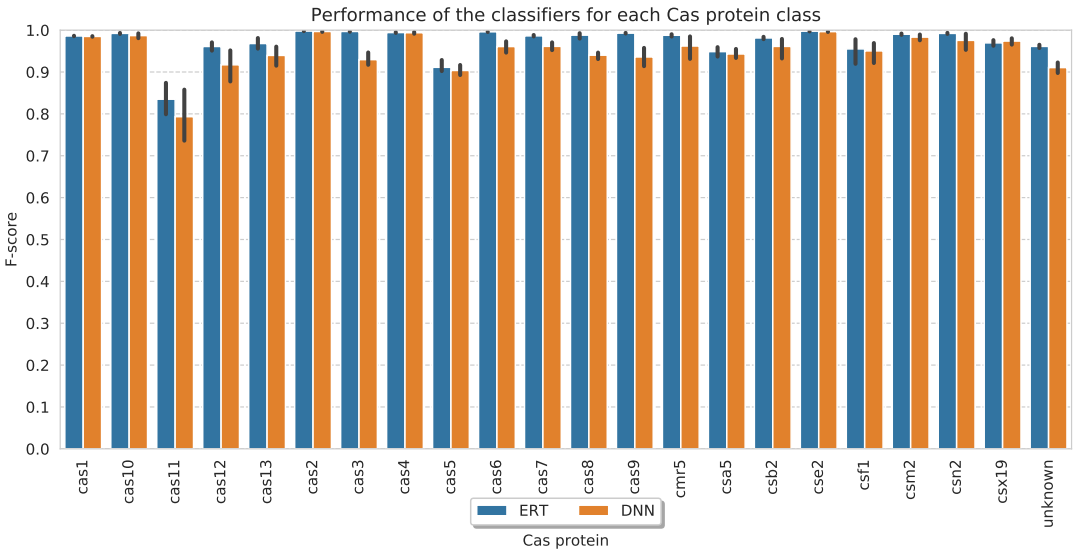


Figure 41 – Cas type prediction F-scores with 1 cas type left out, using the general HMM features.

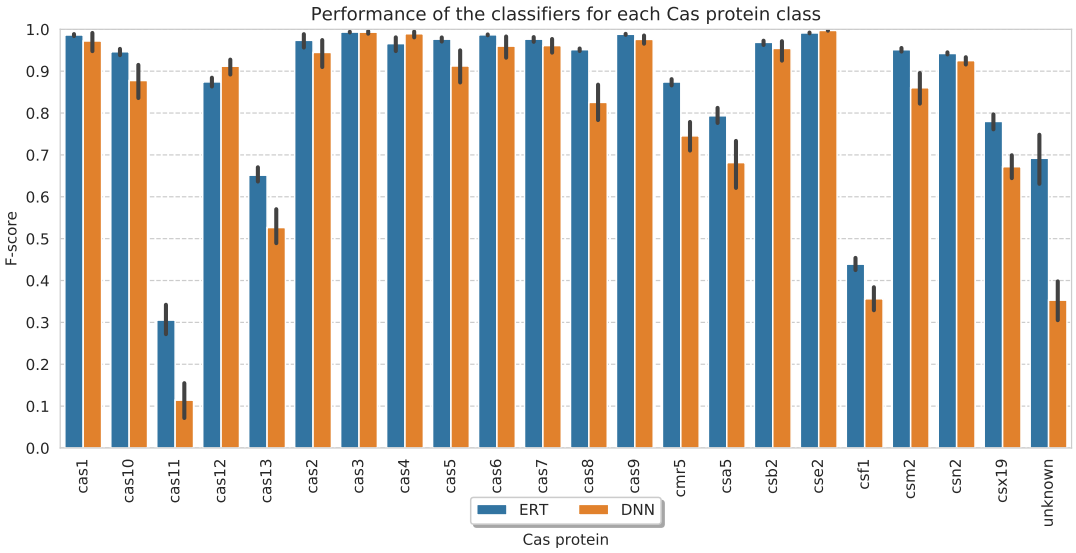


Figure 42 – Cas type prediction F-scores with 3 cas types left out, using the general HMM features.

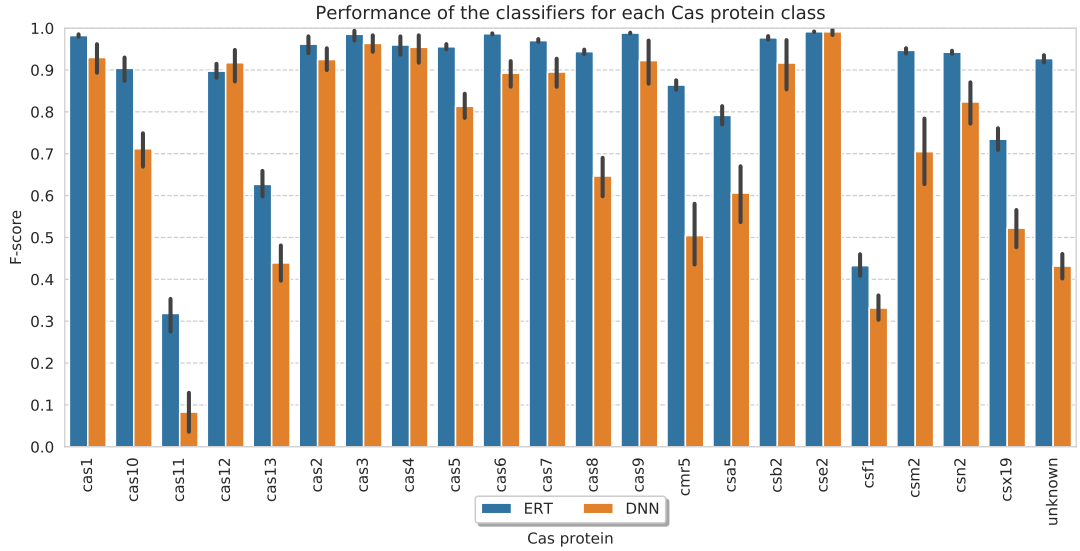


Figure 43 – Cas type prediction F-scores with 1 cas type left out, using a combination of the general HMM and protein properties features.

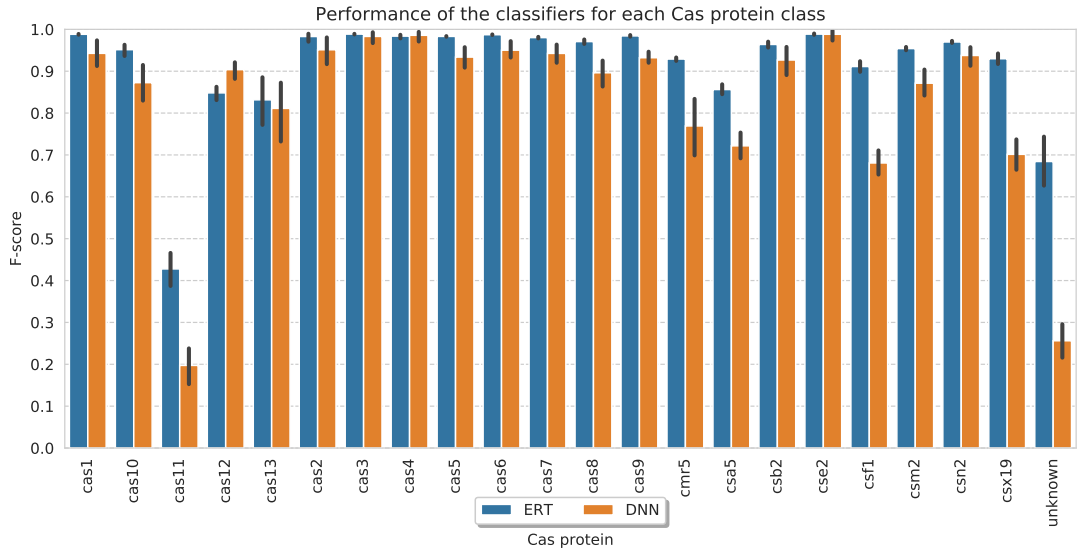


Figure 44 – Cas type prediction F-scores with 3 cas types left out, using a combination of the general HMM and protein properties features.

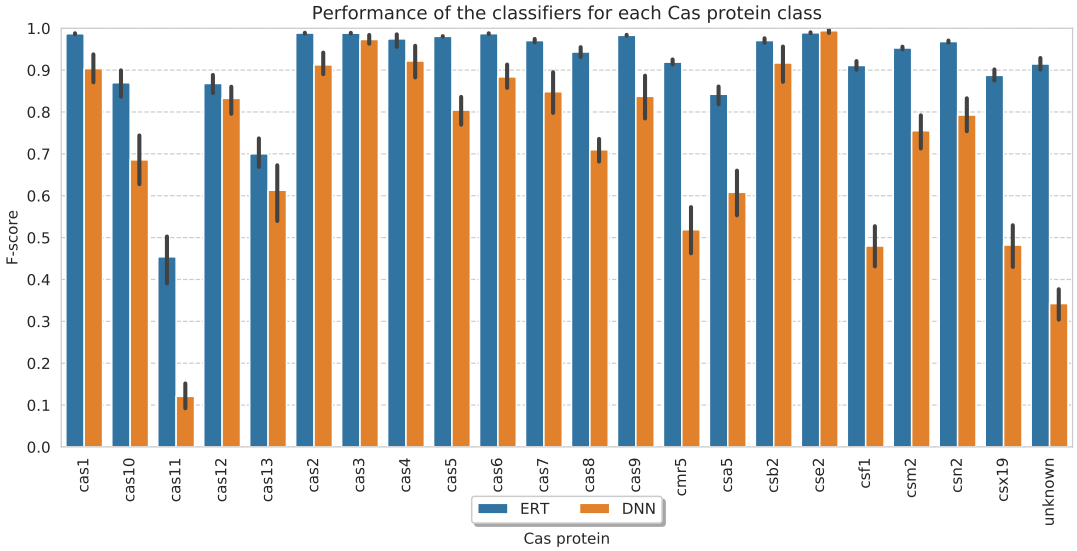


Figure 45 – Cas type prediction F-scores with 1 cas type left out, using the protein properties features.

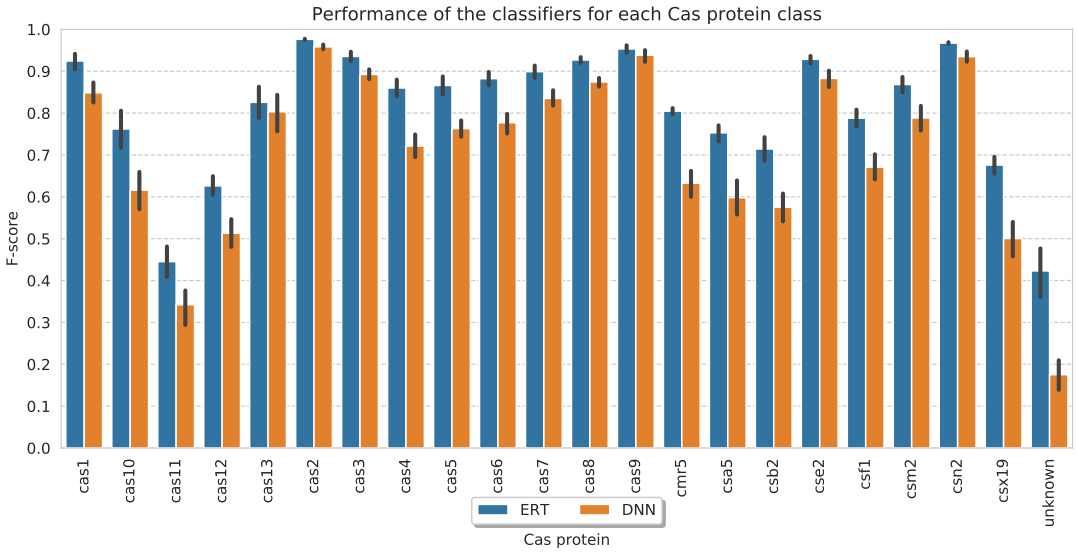


Figure 46 – Cas type prediction F-scores with 3 cas types left out, using the protein properties features.

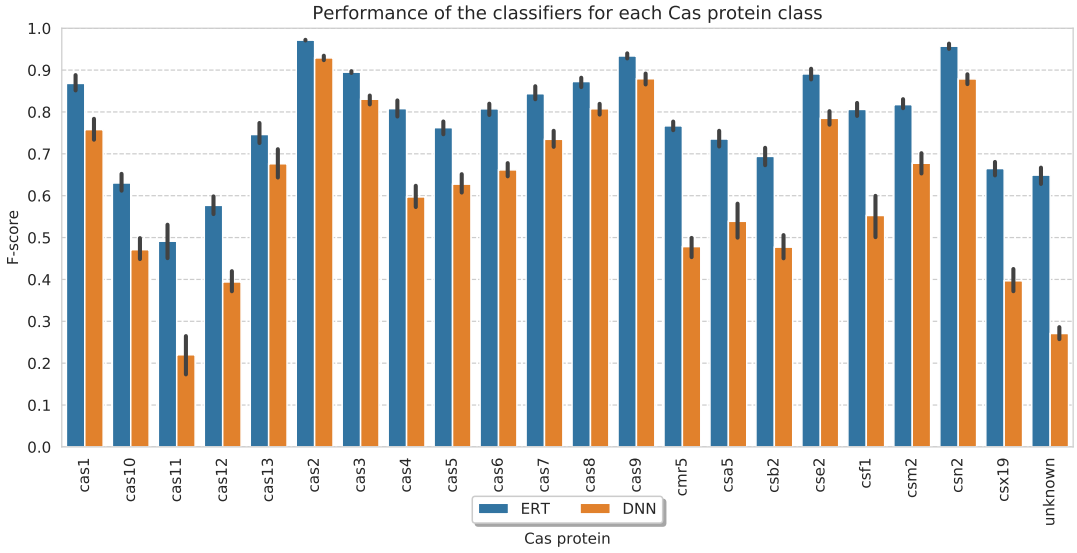


Figure 47 – Phylotree of Cas8 and putative cas8. The tree is generated based on the Neighbour-joining method. Here we showed the distance between the closest 29 proteins from Cas8-family along with the putative cas8 protein.

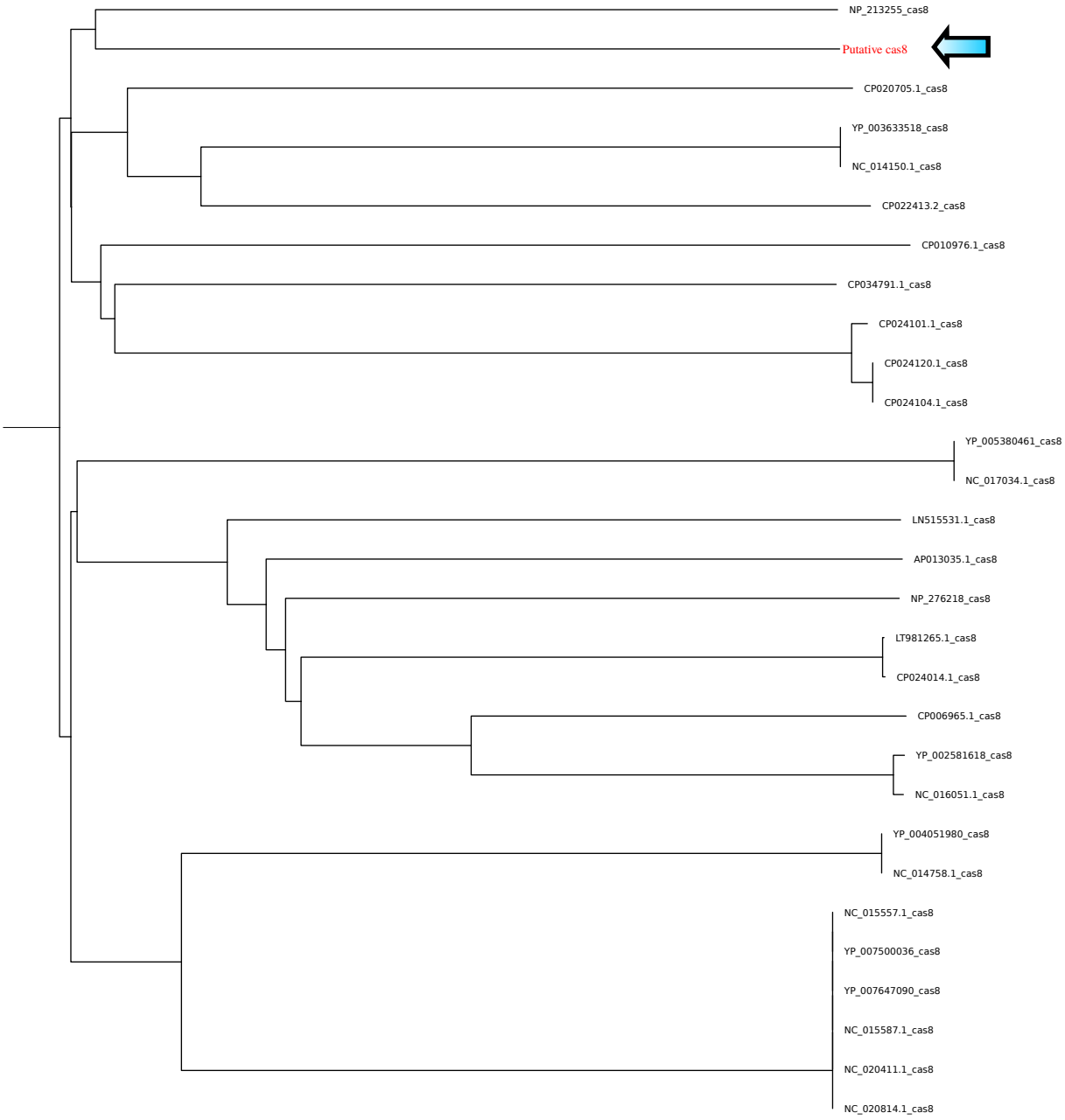


Figure 48 – Multiple Sequence Alignment of Cas8 proteins and the putative Cas8 proteins. The alignment obtained from MUSCLE alignments of 29 Cas8 family and the putative Cas8. The conserved regions are shown on the bottom of the alignment.

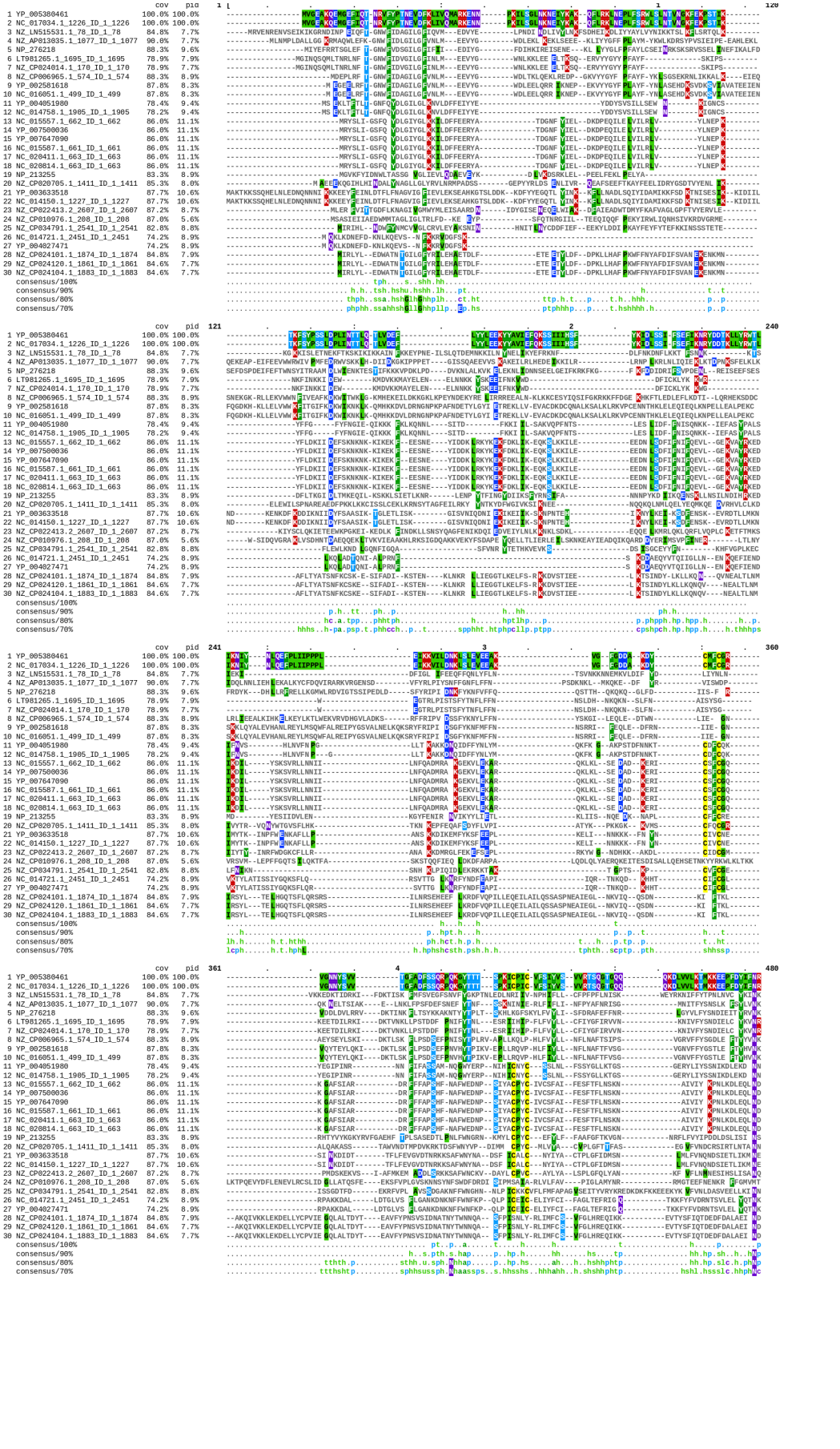


Figure 49 – Multiple Sequence Alignment of Cas8 proteins and the putative Cas8 proteins. The alignment obtained from MUSCLE alignments of 29 Cas8 family and the putative Cas8. The conserved regions are shown on the bottom of the alignment.

Table 31 – Percentage of closest matches for Cas1 under each subtype. The analysis was carried out using the k-nearest neighbors approach with $k = 5$.

	I-A	I-B	I-C	I-D	I-E	I-F	I-U	II-A	II-B	II-C	III-A	III-B	III-C	III-D	III-E	IV-A	V-A	VI-A
I-A	66.59	19.06	0.00	5.65	0.00	0.00	0.24	0.00	0.00	0.00	1.65	2.59	0.94	3.29	0.00	0.0	0.00	0.0
I-B	1.48	94.15	0.15	2.10	0.03	0.03	0.15	0.00	0.15	0.03	0.79	0.38	0.41	0.06	0.00	0.0	0.12	0.0
I-C	0.00	0.26	98.75	0.00	0.00	0.00	0.23	0.00	0.00	0.10	0.31	0.13	0.16	0.05	0.00	0.0	0.00	0.0
I-D	4.48	12.24	0.17	76.55	0.00	0.00	0.00	0.00	0.00	0.00	4.83	1.55	0.00	0.17	0.00	0.0	0.00	0.0
I-E	0.00	0.03	0.01	0.00	99.71	0.02	0.00	0.09	0.00	0.01	0.04	0.06	0.00	0.03	0.00	0.0	0.00	0.0
I-F	0.00	0.10	0.03	0.00	0.00	99.53	0.00	0.03	0.00	0.10	0.03	0.07	0.07	0.00	0.00	0.0	0.03	0.0
I-U	1.54	0.00	20.00	0.00	0.00	0.00	36.92	0.00	0.00	0.00	23.08	13.85	0.00	4.62	0.00	0.0	0.00	0.0
II-A	0.00	0.03	0.03	0.00	0.06	0.03	0.00	96.69	0.00	2.89	0.23	0.00	0.00	0.00	0.00	0.0	0.06	0.0
II-B	0.00	0.49	0.49	0.00	0.49	0.00	0.00	0.49	94.63	0.49	1.46	0.00	0.00	1.46	0.00	0.0	0.00	0.0
II-C	0.00	0.04	0.04	0.00	0.00	0.08	0.00	3.96	0.00	95.38	0.23	0.04	0.08	0.08	0.00	0.0	0.08	0.0
III-A	0.62	4.38	0.68	1.93	0.00	0.00	0.17	1.36	0.06	0.74	79.55	6.02	0.85	3.47	0.06	0.0	0.11	0.0
III-B	2.60	4.25	0.96	1.23	0.14	0.00	0.27	0.00	0.00	0.00	17.67	63.56	0.00	9.18	0.14	0.0	0.00	0.0
III-C	6.67	30.00	0.00	0.00	0.00	3.33	0.00	0.00	3.33	3.33	23.33	1.67	16.67	8.33	0.00	0.0	3.33	0.0
III-D	4.25	2.00	0.25	1.25	0.00	0.00	0.75	0.00	1.00	1.00	26.00	25.25	2.25	34.75	0.75	0.0	0.00	0.5
III-E	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	20.00	0.00	0.00	80.00	0.00	0.0	0.00	0.0
IV-A	0.00	0.00	0.00	0.00	80.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	20.0	0.00	0.0
V-A	0.00	1.76	0.59	1.18	0.00	0.59	0.00	1.18	0.00	0.59	2.35	0.00	1.18	0.00	0.00	0.0	90.59	0.0
VI-A	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	80.00	20.00	0.0	0.00	0.0

Table 32 – Runtime and RAM usage comparison. We selected a set of 650 genomes and achieved the following results using an Intel i5 machine with 8 GB of RAM.

Tool	CPU runtime	RAM usage
Casboundary	7 Hrs 23 Sec	5.3 MB
CRISPRCasFinder	3 Hrs 58 Sec	2.2 MB


ICMC
USP
SÃO CARLOS