FREIE UNIVERSITÄT BERLIN

DOCTORAL THESIS

Design and multi-criteria optimization of cell classifier circuits in cancer therapy

---

Dissertation zur Erlangung des Grades
einer Doktorin der Naturwissenschaften (Dr. rer. nat.)

am

Fachbereich Mathematik und Informatik der Freien Universität Berlin

vorgelegt von:
MELANIA NOWICKA

Berlin, 2023

Betreuerin:	Prof. Dr. Heike Siebert
Erstgutachterin:	Prof. Dr. Heike Siebert
Zweitgutachter:	Prof. Dr. Bernhard Renard
Tag der Disputation:	21.07.2023

FREIE UNIVERSITÄT BERLIN

Abstract

Design and multi-criteria optimization of cell classifier circuits in cancer therapy

MELANIA NOWICKA

Custom frameworks that enable exploitation of incomplete and noisy data reflecting real-world environments are frequently critical to satisfy the needs of a particular synthetic design problem. In this thesis, I focus on the *in silico* design of synthetic circuits for the diagnosis and treatment of cancer. In particular, I develop computational frameworks for the logic-based design of the classifier circuits, utilizing a range of different computational paradigms from machine learning to evolutionary algorithms. First, I focus on the optimization of single-circuit classifiers according to the objectives and constraints imposed by the experimental circuit assembly. I exploit the potential of logic programming, in particular, Answer Set Programming, and propose a workflow for the design of globally optimal logic classifiers. Further, I introduce an alternative, theoretical design of classifiers consisting of multiple circuits, namely, distributed classifiers. I leverage the advantages of ensembles, in particular, collective decision-making, to yield better performance for heterogeneous data. To optimize the ensembles, I develop a custom genetic algorithm, as well as revise the classifier optimality criteria. Next, I focus on refining the evaluation strategies and increasing the robustness of our designs to novel data. Finally, I explore alternative applications beyond cancer classifiers to showcase the versatility of the proposed methods.

Acknowledgements

First of all, I would like to thank my PhD supervisor Heike Siebert for offering me a chance to work on this exciting research topic and for supporting me throughout the PhD journey. I would also like to thank the Discrete Biomathematics group, especially Laura and Elisa.

I would also like to thank colleagues from IMPRS and Max Planck Institute for Molecular Genetics. I had a great time in the programme with you.

I would like to express my gratitude to my Thesis Advisory Committee members, Knut Reinert and Annalisa Marsico, for valuable discussions and their engagement in the role of advisors.

I would like to also thank Bernhard Renard, a member of my Thesis Advisory Committee during his time at FU Berlin, for supporting me during better and worse moments and making academia a better place every day.

I would like to thank the DACS group for the warm welcome during the pandemic and for making my time at HPI amazing. I also thank Marta, Liz, Chris, Katharina, Pascal and others for their support.

I would also like to thank my family and friends, especially Natalia and Martyna, for listening and motivating me.

Finally, I would like to thank my life and work partner, Kuba, for believing in me in moments of doubt and supporting me through the ups and downs. I really appreciate every single sacrifice you made for me to be able to do what I love.

And last but not least, my furry friend Mojra, for purring loudly on my lap while writing this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The new era of synthetic biology

Synthetic biology is transforming modern medicine, advancing material engineering, and supporting the battle against climate change (El Karoui et al., 2019; Voigt, 2020; Roell and Zurbriggen, 2020; Zhao, 2022; An et al., 2023). We are no longer observing an emerging field but approaching the next era of bio-design that leaves the lab and slowly enters our households - for instance, in the form of environmentally-safe synbio-based agriculture (Voigt, 2020; Zhao, 2022). The development of new wet lab techniques provides arrays of tools accelerating the exploration of biological designs in the laboratory (Doudna and Charpentier, 2014; Hsu et al., 2014; Schmidt et al., 2023). Benefiting from this momentum, long-awaited precision medicine based on synthetic biodesign becomes viable, providing novel therapeutics ranging from synthetic enzymes for diabetes to personalized cell therapies for leukemia patients (Voigt, 2020). In particular, different cancer therapies rely on synthetic design to circumvent the cancer escape mechanisms and lower the toxicity of treatment (Nissim et al., 2017; Cho et al., 2018; Angelici et al., 2021). In this work, we focus on synthetic cell classifiers that enable *in vivo* selective targeting of cancer cells (Xie et al., 2011; Mohammadi et al., 2017; Dastor et al., 2018; Angelici et al., 2021). After being delivered to the tumour, the circuits sense a pre-defined array of biomarkers, process them in a logic-inspired manner, and trigger apoptosis in cancerous cells, as briefly illustrated in Figure 1.1. The authors have shown that cell classifiers enable a substantial reduction of cancerous cells, simultaneously displaying minimal toxicity, and thus shape a potential alternative for cancer patients (Angelici et al., 2021).

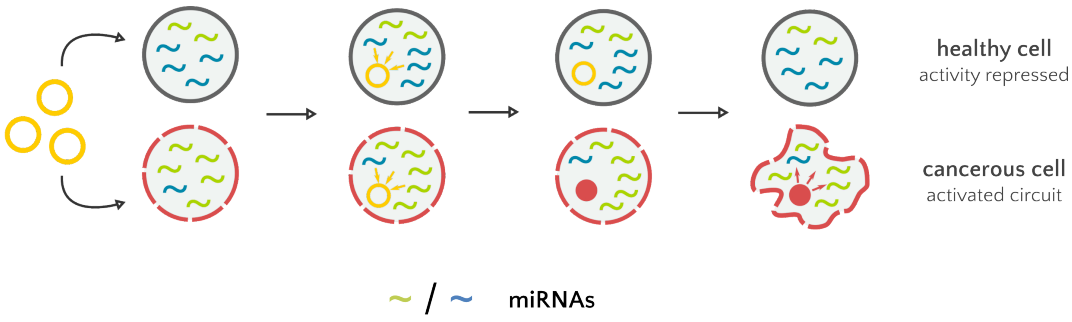


FIGURE 1.1: Classifier circuits (yellow circles) are introduced into living cells (healthy and cancerous), sense different levels of markers enabling distinguishing the cancerous and healthy cells, and activate output production in cancerous cells, causing their apoptosis.

Various diagnostic and therapeutic synbio-based tools are currently under development. However, the design of new biodevices touching upon a wide range of applications requires interdisciplinary effort and close collaboration between the dry and wet labs to achieve success (Ho and Bennett, 2018; Mohammadi et al., 2017; Cubillos-Ruiz et al., 2021; Hérisson et al., 2022; Schmidt et al., 2023; Yeh et al., 2023). As the complexity of synthetic devices often rises with the complexity of the desired functionality, the time and financial resources needed to design sophisticated circuits are increasing. Hence, to mitigate those limitations the vast majority of biological designs are supported by artificial intelligence and automation to save valuable resources, allowing the field to thrive and play a major role in the fast-growing bioeconomy (Zhao, 2022).

The development of synthetic cell classifiers also benefits from computer-aided design, as the vast space of possible circuits hinders the manual search for promising candidates (Mohammadi et al., 2017). However, multiple optimality criteria and design constraints impede the employment of existing methods for classifier design, creating the need for application-tailored approaches. *In silico* screening through potential alternative designs of classifiers and evaluating their performance in different scenarios is crucial for boosting the development of cell classifiers. Mohammadi et al., 2017 proposed a proof-of-concept computational framework for cell classifier design. The authors focus on a particular model of classifiers, assessing its performance in specific cancer case study scenarios. In this work, I propose alternative computational frameworks and circuit architectures, assessing their performance in a broader context. In particular, we develop custom machine-learning approaches for the design and optimization of different cell classifiers, and for estimating their robustness in various environments, including facing novel and heterogeneous data. The following sections outline the background on the development of synthetic cell classifiers for cancer research in the context of basic principles behind synthetic biology. Further, I briefly introduce the primary application of cell classifiers, namely, diagnostics and therapy of cancer. Finally, I outline the motivation behind our work and summarize the main goals of the thesis.

1.2 Designing synthetic devices

Diagnostic and therapeutic biodevices often aim at high precision and steep on-off response in varying environments such as discriminating cell types as healthy or diseased *in vivo*. At the core of the field, synthetic biology relies on engineering principles and aims at rational design or redesign of basic biological parts, more complex circuits, or even entire systems (Endy, 2005). Basic parts such as sensors or actuators can be used to build circuits that allow processing information from the input to the output by sensing certain compounds in the environment and responding with regard to their presence. Examples include biological toggle switches controlling gene expression and production of desired molecules (Bothfeld et al., 2017) as well as cell classifiers - multi-sensor devices that can recognize different cell phenotypes (Xie et al., 2011; Mohammadi et al., 2017). To process the incoming signals, biological circuits often mimic analog and digital computation known from electronic devices and create their biological counterparts in living cells.

However, both digital and analog computing occurs also in natural biological systems, in particular, in the decision-making processes such as development or response to environmental stressors (Daniel et al., 2013) providing a repertoire of existing biological parts and circuitry to design new components and functionalities.

The natural biological systems constantly process various signals, either endogenous or exogenous, and respond according to the computed outcome. For that, they are equipped with complex networks of sensors and actuators that allow reacting to continuous changes. At the cellular level, each cell processes input signals, such as endogenous protein expression levels, that decide its future state. An example is the *lac* operon allowing bacteria to digest lactose (if present) in the absence of glucose (Setty et al., 2003). Briefly, if the lactose level is above a given threshold θ_{lac} and the glucose level is below a given threshold θ_{glu}, then the lactose metabolism can be activated. We can summarize the above-mentioned example with a set of rules resembling pseudocode: if $lactose > \theta_{lac}$ and $glucose < \theta_{glu}$, activate the lactose metabolism. Alternatively, such behaviour can be represented via logic functions, particularly Boolean functions. We can transform the quantitative levels of molecular entities into qualitative categories, e.g., high (1) and low (0), describe the signal processing using logic operators (such as AND or OR) and the decision as True (1) or False (0) meaning, e.g., yes/no or activation/repression. This allows using a Boolean term as a human-readable description of the input-output relation: if *lactose* AND NOT *glucose* is True, then activate lactose metabolism (Benenson, 2009). Note that the complex processing of molecular signals usually cannot be entirely reduced to logic operations, and the Boolean function only approximates a circuit's behaviour. However, it is possible to adjust the biological design to achieve nearly digital computation and simulate the circuit's behaviour in a dry lab (Setty et al., 2003; Didovyk et al., 2015; Mohammadi et al., 2017).

Studying the natural computation in the living cells and whole biological systems, we can (re)design synthetic biodevices using natural and manufactured parts (Meng and Ellis, 2020). Further, we can use the Boolean representation to build schemas of the existing systems and computationally simulate their behaviour. This is particularly valuable in an adjacent research field, systems biology. Here, Boolean representations of the biological networks complement their analysis and accelerate, e.g., drug target discovery (Borriello and Daniels, 2021; Cifuentes-Fontanals et al., 2022). As synthetic biodesign often aims for on-off logic-like circuit behaviour, the Boolean representation can serve not only as an abstract description but also as a baseline for the circuit's design. Hence, various studies presented biodesigns that rely on Boolean logic, from the aforementioned toggle switches (Bothfeld et al., 2017), through multi-cellular frameworks performing Boolean logic (Guiziou et al., 2018; Guiziou et al., 2019) and multi-input cell classifiers (Mohammadi et al., 2017), to different variants of memory-like synthetic circuits (Ho and Bennett, 2018).

Following the engineering principles, multiple 'programming' languages were designed to assist the standardized description and *in silico* design of biological devices (Umesh et al., 2010; Vasić et al., 2020; McLaughlin et al., 2020). Such computer-aided design (CAD) software tools aim at describing the biodevices in a machine-readable manner that allows for modular and scalable design and optimization (Nielsen et al., 2016; Guiziou et al., 2018; Hiscock, 2019; McLaughlin et al., 2020; Gorochowski et al., 2021; Hérisson et al., 2022). For example, Cello (Nielsen et al., 2016) utilizes the programming language Verilog used to model electronic systems to describe the logic of a desired circuit. Recently published Galaxy-SynBioCAD (Hérisson et al., 2022) allows for end-to-end metabolic pathways design. There are no universal tools, and the published frameworks usually cover a particular subset of design problems. Custom approaches must be developed to assess the performance of new design concepts or simply satisfy the constraints of a particular use case. For instance, Mohammadi et al., 2017 developed an application-tailored framework SynNet for cell classifier design and optimization that incorporates desired optimality

criteria and structural requirements of classifiers needed for successful wet lab assembly. As cancer is thus far one of the major health concerns worldwide, there are many available tools for cancer sample classification (Lopez-Rincon et al., 2019; Alharbi and Vakanski, 2023). However, even though they can uncover relevant diagnostic features, they cannot be directly applied to the design of cell classifiers. Thus, none of the previously available tools meets the needs of the respective use-case scenario (Mohammadi et al., 2017).

1.3 Battling cancer

In the past decades, technological advances such as sequencing or novel molecular techniques shed light on the genetic landscape of cancer. The scientific community gathers substantial amounts of data and develops advanced computational approaches, providing extensive knowledge about the mechanisms underlying the disease (Creighton, 2018; Dagogo-Jack and Shaw, 2018). Nevertheless, according to the World Health Organization (WHO), cancer is the cause of one in six deaths worldwide (*WHO report on cancer: setting priorities, investing wisely and providing care for all* 2020) and still poses a considerable challenge for the diagnosis and treatment of patients (Dagogo-Jack and Shaw, 2018; "The global challenge of cancer" 2020), even though new therapeutic agents are approved every year (Dupont et al., 2021). Newly emerging strategies usually target specific types and subtypes of cancer as the complex and dynamic environment of cancer cells impedes the development of versatile and efficient drugs of low toxicity (Dagogo-Jack and Shaw, 2018; "The global challenge of cancer" 2020; Dupont et al., 2021). However, even type-targeting approaches may be unsuccessful facing within-tumour and between-patient diversity of cancer cells (Zhang et al., 2019).

The malignant tumours usually undergo constant stochastic evolution, resulting in cells with heterogeneous genetic and molecular signatures, destabilizing surrounding tissues (Dagogo-Jack and Shaw, 2018). Ultimately, the tumours consist of various cell compositions displaying a range of resistance patterns. The heterogeneity occurs on different levels: intratumoral (within cells belonging to a primary tumour), intermetastatic (between metastases), intrametastatic (within metastatic lesion), and interpatient (Dagogo-Jack and Shaw, 2018; Ward et al., 2021). The interpatient (or intertumoral) heterogeneity occurs between tumours of the same histological type and can impede the between-patient transferability of cancer therapies (Marusyk et al., 2012; Dagogo-Jack and Shaw, 2018). Based on the genetic and molecular characteristics of the patient-derived tumour, we can define subtypes of particular cancers that often demonstrate different treatment resistance behaviour and prognoses (Marusyk et al., 2012). For instance, different therapies are proposed for patients with different breast cancer subtypes (Ward et al., 2021). Another example of patient-personalized treatment is chimeric antigen receptor T cells (CAR T) therapy. It relies on the patient's T cells that are genetically modified outside the subject's organism and delivered back to the patient as targeted therapeutics. This approach was successful in the treatment of patients with acute lymphoblastic leukaemia (Grupp et al., 2013; Dagogo-Jack and Shaw, 2018).

In the case of intratumoral heterogeneity, the cells belonging to the same tumour differ in their genetic and molecular makeup. Thus, sampling the tumour from several sites can result in different cell profiles (Dagogo-Jack and Shaw, 2018; Ward et al., 2021). In such cases, personalized therapeutic agents that use the knowledge

about the genetic composition of the tumour demonstrated better performance facing the heterogeneous environment (Dagogo-Jack and Shaw, 2018). To encompass the diverse cancer environment resulting from inter- and intratumor heterogeneity, synthetic therapeutics that do not account for the tumour- and patient-derived diversity may not be sufficient (Mohammadi et al., 2017). Thus, to circumvent the tumour heterogeneity, novel cancer therapies rely on multi-input biocomputation that takes into account different markers and genetic modalities. Examples include logically programmed immunotherapeutics (Nissim et al., 2017), aforementioned multi-input sensing CAR-T cells (Cho et al., 2018), and, combined with acting at a single-cell level, adenovirus-based cancer cell classifiers (Angelici et al., 2021). The following section focuses on the development of the last of those approaches – the cancer cell classifier circuits.

1.4 Development of cell classifiers

In 2011, Xie et al. proposed multi-input synthetic circuits for the *in vitro* classification of cell phenotypes based on miRNA expression profiles. The authors have demonstrated that the circuits successfully recognize cells belonging to HeLa and control cell lines. Moreover, the circuits selectively trigger apoptosis in HeLa cells, not affecting non-HeLa controls. In another study, Dastor et al., 2018 set up an experimental framework for preclinical assessment of cell classifiers using mouse tumour models. The framework enables validating miRNA candidates and the circuit's ability to yield output production *in vivo*. The authors screened potential input candidates and selected two miRNAs that allow distinguishing cancerous and healthy cells. As an output treatment, the authors chose HSV-TK (Herpes Simplex Virus type 1 Thymidine Kinase) followed by the administration of ganciclovir. This approach is known as gene-directed enzyme prodrug therapy (or suicide gene therapy) and has been previously validated *in vivo* (Wang et al., 2011; Oishi et al., 2022). The circuits are introduced into the cells via an adeno-associated virus (AAV) vector delivery platform. After delivery, the circuits sense the presence of given miRNAs and release an encoded enzyme (HSV-TK), which is followed by the administration of ganciclovir. HSV-TK allows converting the non-cytotoxic prodrug ganciclovir into a cytotoxic compound, subsequently resulting in cell apoptosis (Wang et al., 2011). The authors demonstrated that for the particular mouse setup, a construct built with miR-122 as input and HSV-TK/ganciclovir treatment yields the best performance. This combination allowed for eradicating the vast majority of tumour cells and demonstrated moderate cytotoxicity in healthy cells. However, the authors suggested that multi-input control of drug production could decrease the toxicity of the therapy (Dastor et al., 2018). Thus, they later rearranged the proposed framework using multiple inputs representing different molecular modalities, namely, transcription factors and miRNAs (Angelici et al., 2021). Integration of multiple inputs reduced the therapy's adverse effects and increased the control over circuits, showing promising avenues for clinical translation. As an alternative to miRNAs and transcription factors, classifier circuits can also process promoter activities (Dastor et al., 2018; Angelici et al., 2021). However, regardless of the chosen biomarker type, manual selection of the features and evaluation of classifiers impedes their development. Here, machine learning supports the search for suitable input candidates and simulation of the circuit behaviours, as well as enables exploring alternative circuit designs.

1.5 Machine learning for cell classifier design

To accelerate the design process, Mohammadi et al., 2017 proposed a computational framework allowing searching through a vast space of biomarker combinations to design logic-like cancer cell classifiers. Due to the plethora of feasible solutions, the authors leverage evolutionary algorithms to optimize the classifier's topology. The authors focus on optimizing the classifier topology represented as a single Boolean term (single-circuit classifiers) via *in silico* simulation of the circuit behaviour (see the following chapter for a detailed description). In this work, I developed alternative machine-learning approaches to design logic-based cell classifier circuits. The thesis focuses on four main aspects of classifier design: (1) we leverage logic programming to design single-circuit classifiers based solely on Boolean terms and obtain globally optimal classifiers, (2) we introduce a new, theoretical topology of a classifier circuit consisting of multiple single-circuit classifiers and develop a proof-of-concept genetic algorithm for their optimization, (3) we increase the classifiers' robustness to data heterogeneity and improve the evaluation strategy to assess classifier susceptibility to noisy data, (4) finally, we extend our methods to create a reusable framework and show-case alternative applications.

The thesis is structured as follows. Chapter 2 contains preliminaries on the state-of-the-art biological and computational design of cancer cell classifiers comprising the data and optimality criteria. In Chapter 3, I present the logic programming-based workflow for the optimization of single-circuit cell classifiers based solely on discrete data. In Chapter 4, I introduce a theoretical design of multi-circuit cell classifiers and propose a proof-of-concept genetic algorithm to optimize such circuits. In Chapter 5, I present a refined optimization strategy of the multi-circuit classifiers and the evaluation scheme to assess the robustness of classifiers in different scenarios. Chapter 6 contains exemplary extensions and applications of the presented computational approaches beyond the cancer therapeutics scenario. Finally, Chapter 7 summarizes my contributions, discusses the limitations, and outlines the future of the presented work.

Chapter 2

From data to circuits

2.1 The inputs: miRNAs

The design of synthetic cancer cell classifiers begins with data describing cancerous and non-cancerous cells. Cell classifier circuits developed so far can process miRNA expression levels, promoter activities and transcription factors as input signals (Dastor et al., 2018; Angelici et al., 2021). The circuits proposed by Xie et al., 2011 and Dastor et al., 2018 can sense miRNAs (microRNAs), small non-coding RNAs consisting of ~22 nucleotides. miRNAs negatively regulate gene expression in the process of RNA silencing and post-transcriptional regulation by base-pairing to the complementary sequences occurring in mRNA, which would normally be further translated to proteins. By targeted binding to the mRNA sequence, miRNAs repress the translation of a given protein to regulate its expression in a cell (Bartel, 2009; Lan et al., 2015). The post-transcriptional regulation mechanism enables control of many cellular processes such as proliferation, apoptosis, or stress responses (Lan et al., 2015).

Hundreds of miRNAs regulate protein expression in humans (Lan et al., 2015). Interestingly, a single miRNA can regulate the translation of many mRNAs, and a single mRNA can be a target of several miRNAs (Hashimoto et al., 2013). As shown in Figure 2.1a if a miRNA targeting a given mRNA is present in the cell, it can bind to the complementary part of the mRNA sequence and repress its translation. Otherwise, the protein is synthesized (Figure 2.1b). The efficiency of protein expression is regulated by the expression level and activity of the particular miRNAs in a cell. Abundant or highly expressed miRNAs block the protein translation with potentially higher efficiency than the lower-expressed ones (Bartel, 2009). Hence, miRNAs can act as switches that turn protein expression on and off or regulate the protein expression in a cell to maintain its desired level (Bartel, 2009).

Overall, miRNAs regulate many different cellular processes, from growth, through differentiation to cell death. Differential expression of miRNAs was described in different cell types, as well as under different cell conditions (Lan et al., 2015; Ludwig et al., 2016). In particular, miRNAs were shown to be differentially regulated between cancerous and non-cancerous cells (Lan et al., 2015; Iorio and Croce, 2012). Thus, differentially expressed miRNAs provide a valuable source of information about tumour development, progression, and response to a therapy (Lan et al., 2015; Iorio and Croce, 2012) and have been considered diagnostic and prognostic biomarkers in cancer.

2.2 The data: miRNA expression profiles

Various databases gather information about miRNA regulation in cancer, e.g., The Cancer Genome Atlas or dbDEMC (Yang et al., 2017b; Sarver et al., 2018). The Cancer

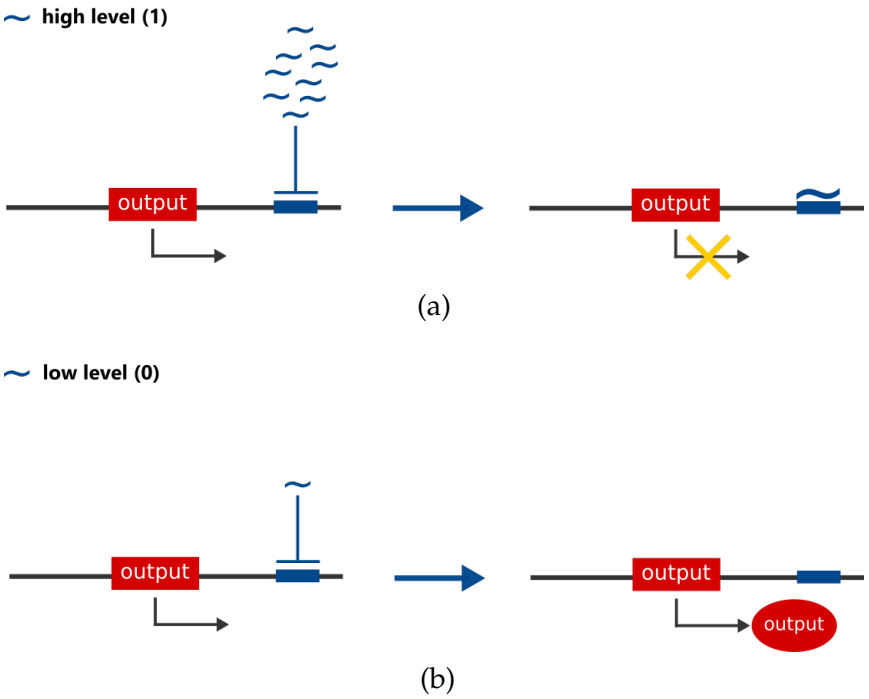


FIGURE 2.1: Different levels of miRNA regulate protein expression. (a) Highly expressed miRNA base-pair to the complementary part of the mRNA sequence, and the expression of an output protein is repressed. (b) Low-expressed miRNA does not bind to the mRNA sequence, and an output protein is synthesized.

Genome Atlas (TCGA) contains diverse data, including genomic, transcriptomic and proteomic data sets. dbDEMC gathers data sets of differentially expressed miRNAs in human cancers obtained mainly from microarray experiments deposited in Gene Expression Omnibus (GEO) (Yang et al., 2017b). Both databases allow accessing multiple data sets allowing for comparative analysis of miRNA expression across cancerous and non-cancerous tissues (Li et al., 2008; Burchard et al., 2010; Jang et al., 2012).

ID	ANNOT	miR-a	miR-b	miR-c	miR-d
1	0	28	856	567	233
2	1	235	56	234	256
3	1	1978	349	23	198

FIGURE 2.2: Example of a simplified miRNA expression data set. The first column (ID) contains unique sample IDs, and the second (ANNOT - annotation) sample assignment to a particular class (cancerous - 1, control - 0). The IDs and annotation are followed by miRNA expression profiles given as, e.g., sequencing read counts.

Regarding cell classifier design, miRNAs that are differentially regulated, i.e., up- or down-regulated in the cancerous cells compared to the control, are candidates for inputs (Xie et al., 2011; Mohammadi et al., 2017). An exemplary data set

containing miRNA expression data is presented in Figure 2.2. The first column includes unique sample IDs, and the second is the sample's assignment to a particular class, where 0 is a label indicating the negative (healthy) and 1 indicates the positive class (cancerous). The following columns are miRNA expression profiles describing the miRNA regulation among the samples. Depending on the employed technology, the measured expression is stored, e.g., as sequencing read counts or signal intensities. Various approaches to the analysis of differential expression data are available (Robinson et al., 2009; Soneson and Delorenzi, 2013). Data discretization, described further in the next section, is an approach enabling obtaining clear-cut information about the miRNA expression level while being compatible with the logic design of cell classifiers.

2.3 Discretization

Discretization enables transforming quantitative into qualitative data by mapping the continuous values into a finite number of non-overlapping states according to a given threshold. In terms of expression data, the continuous values are discretized into states indicating expression levels, e.g., high or low (Gallo et al., 2016b). This technique is a common step in preprocessing biological data to uncover the patterns underlying particular biological mechanisms and find features that describe the differences between samples (Gallo et al., 2016b; Wang et al., 2014). Discretization provides clear-cut and easily interpretable information about the expression levels and makes learning from the data more efficient. Further, the biological and technical noise in the expression data can be partially absorbed in the discretization process leading to better performance (Gallo et al., 2016b). However, the significant reduction in data resolution also poses a risk of potential information loss. Therefore, the discretization approach should be applied carefully and account for potential resolution loss.

From the technical perspective, discretization can be supervised or unsupervised, i.e., considering the sample labels or not, respectively. In the process, the continuous values are mapped into two (binary), three (ternary) or many categories (multi-level discretization). Also, the discretization can be applied to different data scopes, such as a single data point, sample, feature, or the entire data set (Gallo et al., 2016b).

ID	ANNOT	miR-a	miR-b	miR-c	miR-d
1	0	28	856	567	233
2	1	325	56	234	256
3	1	1978	349	23	198

e.g., θ = 300

ID	ANNOT	miR-a	miR-b	miR-c	miR-d
1	0	0	1	1	0
2	1	1	0	0	0
3	1	1	1	0	0

FIGURE 2.3: Example of data discretization according to a discretization threshold θ.

In this work, we focus on discretizing the expression data into two states, i.e., binarization. We define a binarized data set $D = (S, A)$ as a finite set of samples $S \subseteq \{0,1\}^m$, where $m \in \mathbb{N}$ is the number of miRNAs and $A : S \longrightarrow \{0,1\}$ is sample annotation. miRNAs are binarized into two states according to a given threshold θ: up- (1/high) if the miRNA expression value is $\geq \theta$ and down-regulated (0/low) if the expression value is θ. We consider two different data scopes: the threshold is either identical for the entire matrix or thresholds are assigned to particular miRNAs.

A general example of discretization using a single threshold for the entire matrix is shown in Figure 2.3. In the latter case, θ is a vector of thresholds $\theta_i \in \mathbb{R}$, $i \in \mathbb{N}$, and $i \in [1, 2, ..., m]$. Note that θ_i is usually a value higher or equal to the minimal expression and lower or equal to the maximal expression of miRNA i in the dataset D. As a result, miRNAs are high (1) or low (0) in particular samples, irrespective of the data scope. A miRNA is non-regulated if its state is either 0 or 1 for every sample (e.g., Figure 2.3, *miR-d*). As such a miRNA is noninformative regarding sample classification, it can be removed from further analysis. Some miRNAs can correctly separate the samples into the two categories implied by the annotation (e.g., Figure 2.3, *miR-a*). However, biological data is usually affected by technical artifacts, and thus, we expect most of the miRNAs to separate the samples only partially.

2.4 Boolean representation of a circuit

A single classifier circuit can be represented by a boolean function $f : S \longrightarrow \{0, 1\}$, where logic inputs correspond to miRNA expression levels defined as low (0) or high (1) and the output to the cell condition classified as non-cancerous (0) or cancerous (1). Regarding the biological function, the binary output equal to 1 indicates the circuit activation and 0 - the repression of output production. Due to the wet lab assembly limitations, the function must be given in the *Conjunctive Normal Form* (CNF) (Mohammadi et al., 2017). This representation corresponds to the combination of the available biological building blocks. A function in the conjunctive normal form is a conjunction (logical AND) of clauses where each clause is a disjunction (logical OR) of literals. The literals can be negative or positive corresponding to negated (logical NOT) or non-negated atoms (inputs), respectively. In terms of synthetic circuit design, the literals correspond to the miRNA inputs and the clauses to the biological modules concatenated into a larger logic function. Ultimately, according to Mohammadi et al., 2017, the function can comprise up to 10 miRNAs and 6 gate modules combined with AND gates ($\wedge$). Mohammadi et al., 2017 proposed two modules that fulfil the assembly restrictions according to the proposed model: an OR gate module ($\vee$) comprising up to 3 miRNAs and a NOT gate module ($\neg$) with a single miRNA. In terms of gates, the circuit can consist of up to two OR gate modules and up to four NOT gate modules. The following term:

$$(miR\text{-}a \vee miR\text{-}b) \wedge \neg miR\text{-}c \tag{2.1}$$

is an example of a 3-miRNA classifier consisting of 2 gate modules, one OR module and one NOT module. The function outputs 1 (activation) if either *miR-a* or *miR-b* (or both) is up-regulated (1) and *miR-c* is down-regulated (0). Otherwise, the output production is repressed (0).

2.5 A mechanical model of the circuit

For the purpose of computational modelling and optimization of classifier circuits, Mohammadi et al., 2017 developed a mathematical model that enables estimating the level of an output compound for a given circuit based on a Boolean term representing the circuit's topology, a vector of continuous-valued miRNA levels corresponding to the circuit's inputs, and a pre-optimized set of biochemical parameters. The model comprises Hill equations enabling the simulation of the production of a

desired output with regard to the given levels of miRNAs. Below a certain threshold of the output compound, the cell should survive, while above the threshold, it should undergo apoptosis. In terms of the therapeutic application, the lower is the output concentration in healthy cells, the less toxic the treatment is. Further, a larger margin between the outputs in the positive and negative classes increases the treatment's accuracy. Ideally, the output in healthy cells is close to zero, and the output in cancerous cells is sufficient to cause apoptosis without triggering adverse side effects.

In case of expression levels that are very high or very low, the model can be well approximated by a Boolean function (Mohammadi et al., 2017). Figure 2.4 illustrates a schema of the biological classifier circuit and a corresponding Boolean topology (matching the Boolean term introduced in 2.1).

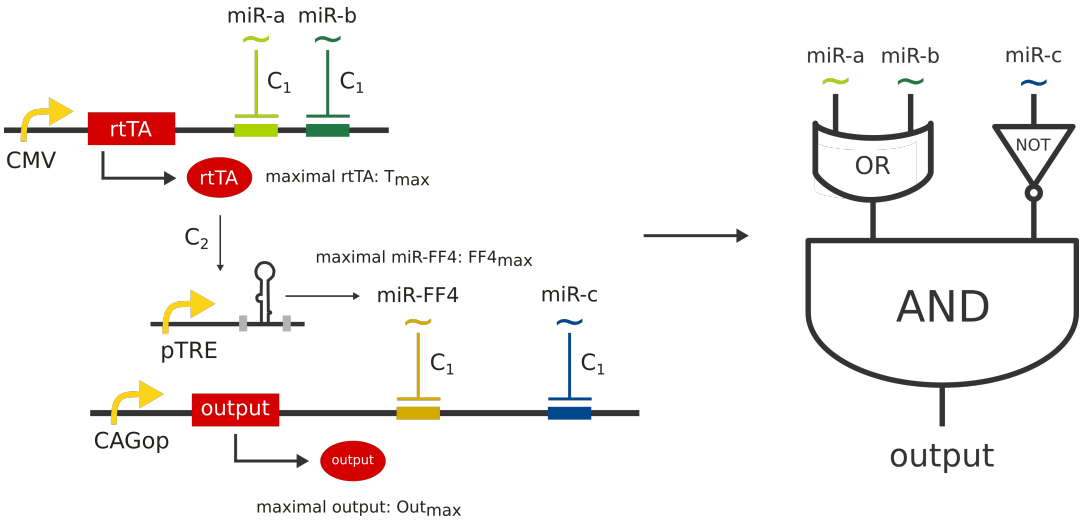


FIGURE 2.4: A schema of the circuit (left) and its Boolean representation (right). Adapted from Mohammadi et al., 2017.

The model comprises five biochemical parameters ($C1$, $C2$, T_{max}, $FF4_{max}$, Out_{max} shown in Figure 2.4) optimized by the authors to maximize the margin between the output released in cancerous versus control cells (further described in S2.1). Each parameter set corresponds to a discretization threshold that can be used to binarize the expression data. The circuit's design utilizes the inherent miRNA function to base pair to complementary sequences of the output sequence and regulate its production. According to the schema, NOT gate modules (miR-c, Figure 2.4) block the output compound production when the concentration of the corresponding input miRNA is high in a cell. Thus, only the miRNAs which are low-expressed in the diseased cells can be candidates for inputs in the NOT gate module in order to promote the production of the output compound in the tumour. Unlike the NOT gates, the OR gate modules (miR-a OR miR-b, Figure 2.4) control the output production indirectly via an artificial miRNA miR-FF4, which acts as an additional NOT gate. Here, the production of the output compound is promoted via the prevention of miR-FF4 activation. This can be achieved by blocking the production of the rtTA activator by miRNAs occurring in the diseased cells at higher concentrations. Thus, only the miRNAs which are highly expressed in the diseased cells may be used as inputs in the OR gate. The model is explicitly employed by Mohammadi et al., 2017 to design and optimize classifier circuits. The following section outlines the basics of the computational framework proposed by the authors.

2.6 Computer-aided design of cell classifiers

Mohammadi et al., 2017 proposed an evolutionary algorithm (EA) that enables the optimization of the circuit's topology regarding the output concentration in cancerous and non-cancerous cells. Evolutionary algorithms are problem-independent population-based metaheuristics inspired by the process of natural selection occurring in biology (Mitchell, 1996). In the Darwinian theory, a population of individuals undergoes development, adapting to the surrounding environment through selective reproduction and survival. In nature, different biological processes, such as genetic recombination or mutation, contribute to the diversity of the population. Among individuals variously adapted to the environment, the process of natural selection favours the fittest individual to be a part of the next generation. In terms of an evolutionary algorithm, a population of candidate solutions (individuals) evolves through the algorithm's iterations, undergoing alterations and producing new populations until the termination criterion is met.

EAs are often employed to solve complex search problems in which the explosion of feasible solutions impedes using deterministic approaches. Hence, the authors employ an EA to optimize the topology of the classifier and propose two variations of the algorithm: a specific and a general search differing mainly in terms of the optimized objectives. The specific search employs the previously described biochemical model and parameters to simulate the real-valued circuit's concentrations. In contrast, the general search does not rely on a biochemical model but uses solely the Boolean approximation of the circuit. As previously mentioned, the biochemical model enables the estimation of the circuit's output concentration in relation to the real-valued miRNA levels. In practice, circuits that operate under clear-cut, biologically viable thresholds are challenging to manufacture. Thus, proper *in silico* evaluation of the circuits is crucial for their success.

To assess the circuit's performance, Mohammadi et al., 2017 proposed two objective functions in a hierarchical relation that enable *in silico* estimation of the circuit's performance. The primary score, the area under the ROC curve (S_{AUC}), measures the classifier's accuracy under different decision thresholds that separate the positive and negative classes. If more than one classifier performs equally well, the authors employ an additional score to select preferred candidates. The S_m score comprises two margins separating positive and negative samples: the average margin (M_a) that measures the difference between the average output in the positive and negative sample class and indicates how well the output concentration separates all samples in the dataset. The worst margin (M_w) is the smallest output margin between the pair of the closest (in terms of the circuit's output) positive and negative samples and helps to capture the outliers. The two margins are combined into a weighted sum:

$$S_m = \lambda M_a + (1 - \lambda) M_w \tag{2.2}$$

where $\lambda \in [0, 1]$ is a weight that specifies the margin's contribution to the score. In the following chapter, we employ the described scores to evaluate and compare the prediction accuracy of our framework and the approach proposed by Mohammadi et al., 2017.

As an alternative to the specific search, the authors propose a general search approach that does not rely on the pre-optimized biochemical parameters. The general search enables the optimization of the classifiers based on their Boolean topology.

First, the authors compute so-called binary margins as a logarithmic distance of the continuous input expression from the pre-defined binarization threshold. Then, the binary margins are propagated through the circuit from the input to the output according to pre-defined rules. In the end, the output margin is employed to compute the average and worst classification margin that corresponds to the margins computed for the specific search. For more details, please refer to Mohammadi et al., 2017. The authors compare the performance of the specific and general search using the biochemical model and conclude that the general search does not perform substantially worse than the specific search in terms of the real-valued classification of samples.

Besides the above-described scores, the authors implement a so-called pruning procedure. As the difficulty of the circuit's assembly rises together with its size, the authors shorten (if possible) the optimized circuits while aiming to maintain their accuracy. Thus, miRNA inputs that only minimally contribute to the classifier's performance can be removed to decrease the complexity of a circuit. Both algorithms return Boolean terms as the circuit's representation.

Chapter 3

Discrete single-circuit cell classifiers

Contribution Note

Part of the work presented in this chapter has been published as a joint article by Katinka Becker (KB), Hannes Klarner (HK), Melania Nowicka (MN) and Heike Siebert (HS) in Becker et al., 2018.

KB, HK and MN contributed equally to this work. KB, HK, and HS conceived the study. If not stated otherwise, KB and HK implemented the code presented in the publication and this chapter. HK designed and performed the simulated data case studies. MN describes the work done by KB and HK for the purpose of the study outline and discussion. MN extended the framework with step-wise constraint relaxation optimization and the evaluation procedure, added post-processing steps, as well as performed the breast cancer case studies presented in the thesis. The code is deposited at: https://github.com/hklarner/RnaCancerClassifier.

The breast cancer data and the assigned discretization thresholds were received from our collaborators at ETH Zurich: Yaakov Benenson, Niko Beerenwinkel and Pejman Mohammadi. The data were binarized according to thresholds proposed by the authors and formatted by MN. Note that the development of both studies (by Mohammadi et al., 2017 and Becker et al., 2018) overlapped in time.

A comparison of binary and continuous settings (presented at the end of the chapter) was performed by MN and is a result of a meeting with Yaakov Benenson and Nico Beerenwinkel at ETH Zurich in Basel in February 2018.

3.1 Introduction

As the classifiers' topologies can be represented as Boolean terms, mathematical modelling offers a variety of tools to support their *in silico* design. Nonetheless, the Boolean framework, in particular, logic programming, remains largely unused in the context of synthetic circuit design (Xie et al., 2011; Moon et al., 2012; Mohammadi et al., 2017). Although Mohammadi et al., 2017 proposed a general search algorithm that relies on the Boolean approximation of a circuit, it is embedded in a metaheuristic approach without explicitly relying on the logic toolkit. In our work, we show the potential of formal methods, particularly Answer Set Programming (ASP), in the

context of cell classifier design. Answer Set Programming is a declarative problem-solving paradigm (Kaufmann et al., 2016) which, in contrast to imperative programming, allows defining only the computational logic of the problem without explicitly specifying a strategy to solve it. The advantage of ASP is that usually, difficult search problems can be solved efficiently without employing substantial computational resources (Kaufmann et al., 2016). Also, ASP has already been applied to solving various biological questions (Gebser et al., 2010; Guziolowski et al., 2013; Palu et al., 2018). For instance, the BioASP software collection (Gebser et al., 2010) gathers multiple ASP-based tools applied in systems biology. Furthermore, the solver can search through the entire search space in a short time while enumerating all globally optimal solutions or proving that no solution exists for a particular problem. However, vast and complex data sets may increase the run-time and require additional resources or heuristic approaches (Guziolowski et al., 2013; Kaufmann et al., 2016).

In this chapter, we employ Answer Set Programming to design cell classifiers. We focus on the Boolean representation of the classifiers and exploit the advantages of the ASP toolkit to find globally optimal logic terms ensuring the biological feasibility of the classifiers. In this context, a classifier is represented solely by a Boolean term that outputs the binary cell state, given a discretized miRNA profile as input. We develop a multi-step workflow for optimizing the classifiers with regard to their accuracy and size and evaluating the designs employing different performance metrics, including the scores proposed by Mohammadi et al., 2017 (described in 2.6). We tailor our implementation to distinguish between healthy and cancerous cells based on miRNA expression profiles, although the proposed workflow can be applied beyond the described scenario.

The following sections describe the employed data, proposed workflow and the classifier optimization strategy. Next, we present simulated data studies estimating the method's performance in terms of generalization error and scalability. To showcase the performance of our approach on real-world data, we present breast cancer case studies and compare the results of the ASP-based strategy with the state-of-the-art method proposed by Mohammadi et al., 2017. Here, we compare the classifier's performance in two settings: using binary and non-binary performance metrics. Finally, we discuss the limitations of the Boolean representations of biological circuits as well as the shortcomings of the employed approach and future outlook.

3.2 Methods

3.2.1 Answer Set Programming workflow

As previously mentioned, the ASP-based problem-solving approach comes with computational advantages. However, encoding the problem as an ASP program requires familiarity with the ASP semantics. A simplified Answer Set Programming solving pipeline is presented in Figure 3.1. First, the problem must be modelled as a logic program according to the mentioned ASP semantics. This comprises the data, problem-imposed constraints and the problem's logic. Then, the program is grounded, i.e., all variables occurring in the program are replaced by variable-free terms. Finally, the solver takes the propositional (variable-free) program as input and computes so-called answer sets, namely, the solutions (Kaufmann et al., 2016). Manual encoding of large data sets and alternating constraints into the ASP language repeatedly is inefficient, impeding its usability. Thus, we automated the encoding of the data and user-specified constraints into an ASP program as well as developed an objective function described by a set of logic rules. The program is later

grounded by an ASP grounder and solved by a solver. Various ASP-solving software is available (Febbraro et al., 2011; Gebser et al., 2012; Kaufmann et al., 2016). To ground and solve cell classifier programs, we employed the grounder *gringo* and the solver *clasp*, both part of the Potsdam Answer Set Solving Collection *Potassco* (Gebser et al., 2012; Kaufmann et al., 2016). The *Potassco* project offers a well-maintained and accessible collection of tools designed for a wide range of solving tasks. Finally, the returned classifiers are evaluated according to the performance metrics.

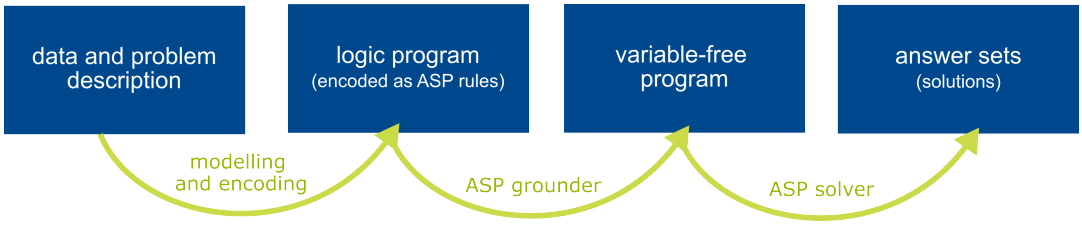


FIGURE 3.1: The Answer Set Programming pipeline.

Figure 3.2 presents a visual description of the particular steps of our workflow. First, the data, classifier constraints, e.g., specifying the wet lab assembly feasibility, and the optimization strategy are translated into an ASP program. The workflow takes as input the binarized training data in a format presented in Section 2.3 and constructs classifiers according to the user-defined constraints imposed on the Boolean function, for instance, by the lab assembly of the circuits (described in 2.4). The user can also specify, e.g., whether the miRNAs must be unique in the classifier or repetitions of the same miRNAs are allowed. More details on the implementation can be found in the code and data repository. Then, the program can be grounded with *gringo* and solved with *clasp*. Returned classifiers are assessed with a two-step evaluation procedure: using binary metrics and the evaluation scores proposed by Mohammadi et al., 2017. Based on this assessment, we identify the best-performing classifiers. The details regarding the optimization strategy and evaluation approach are described in the following sections.

3.2.2 Optimization strategy

In Section 2.6, we describe two optimality criteria related to the classifier design: accuracy and size. Motivated by the medical application, we optimize those criteria hierarchically: prediction accuracy is prioritized over the size of classifiers, similarly to the strategy proposed by Mohammadi et al., 2017.

The optimization of accuracy is embodied within the program's constraints. Here, the bounds on allowed errors (i.e., the maximal number of misclassified samples) must be specified beforehand by the user. The workflow enables searching for *perfect classifiers*, namely, classifiers that distinguish all the samples in the data without errors (by applying the *perfect classifier* option or setting upper bounds on errors to 0). However, most biomedical data, particularly expression data, is exposed to noise derived from experimental artefacts or data pre-processing (Whalen et al., 2022). Thus, we introduce an option to search for so-called *imperfect classifiers* by setting upper bounds on the number of allowed false-positive (FP) and false-negative (FN) errors. The division provides higher flexibility regarding the classifier's application as a particular error type can influence the efficiency (false negatives) versus toxicity balance (false positives). The workflow allows enumerating all globally optimal solutions for a given set of constraints. If no solutions are found for the particular

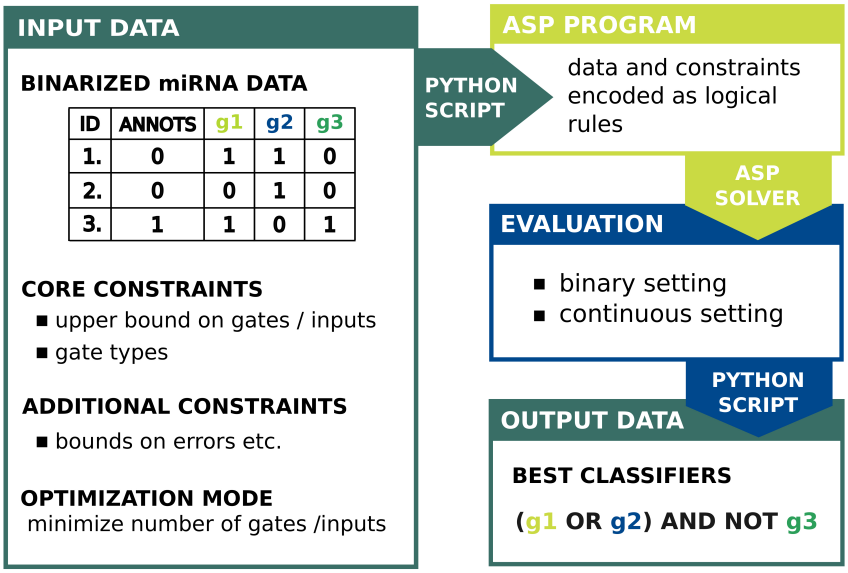


FIGURE 3.2: All steps of our workflow. First, the data and user-specified constraints are encoded as an ASP program. Next, the program can be solved with the *clasp* solver. The returned classifiers are then evaluated. Figure created by MN.

error restrictions, the user can relax the constraints and repeat the optimization procedure. Note that relaxing upper bounds on the number of errors too far can lead to an explosion in the number of returned solutions.

Different combinations of allowed FPs and FNs can result in equally precise solutions in terms of the total number of errors, e.g., 4 FPs and 3 FNs, as well as 3 FPs and 4 FNs resulting in 7 errors in total. Thus, we employed a step-wise *constraint relaxation* that allows optimization of the number of allowed errors and enumeration of globally optimal solutions for all the combinations. First, we search for a *perfect classifier*. Then, if such does not exist, the constraints are step-wise relaxed, allowing up to 1 FN and 0 FPs, 0 FNs and 1 FPs, and 1 FN and 1 FN, examining all the combinations of allowed errors up to the pre-defined bounds.

In terms of size optimization, the following strategies are included in the workflow: (1) minimize the total number of inputs, (2) minimize the total number of gates, (3) minimize the number of inputs followed by the number of gates, and (4) minimize the number of gates followed by the number of inputs. (1) and (2) enable the minimization of the number of inputs and gates in the Boolean term, respectively. The two last strategies, (3) and (4), are hierarchical, bi-level optimization problems, i.e., the upper-level problem is solved first, followed by the lower-level problem. Note that different optimization strategies can result in different solutions for the same data and constraints.

3.2.3 Post-processing

The Potassco solver allows for isomorphic classifiers to be counted as separate solutions. The gates and inputs in each returned function are ordered, e.g., gates are assigned an integer identifier determining which input belongs to which gate. Permutations of the assigned identifiers result in symmetric (or isomorphic) solutions

differing solely in their arrangement. Since the solver enumerates all the optimal solutions, returning isomorphic functions can impede the computation by significantly increasing the run time. However, in our application, we can solve this problem through a post-processing step by comparing the returned solutions and choosing one representative per isomorphic class (details described in Section S3.1). We present an exemplary case study in Section 3.2.6. The procedure was implemented by MN.

3.2.4 Classifier Evaluation

We distinguish two settings for the classifier performance evaluation. In the Boolean setting, we evaluate how well the logic term separates samples for a given binarized dataset employing the false positive ($FPR = FP/N$) and false negative rates ($FNR = FN/P$). FPR allows estimating the probability of classification as a false positive, i.e., classifying a control sample as cancerous, while FNR estimates the probability of a false negative, i.e., classifying a cancerous sample as healthy. In the continuous setting, we adopt the scores proposed by Mohammadi et al., 2017 and described in detail in Section 2.6. Employing these measures enables estimating whether Boolean classifiers optimized within the Answer Set Programming paradigm result in a similar performance in terms of sample separation as the classifiers presented by Mohammadi et al., 2017. If the data size permits, we perform a 3-fold cross-validation to test the generalization power of the computed classifiers facing unseen data.

3.2.5 Simulated data generation

To assess the performance of our approach, we simulate synthetic data sets reflecting two different scenarios. We either: (1) ensure a feasible solution exists or (2) create a data set without a known solution. First, we simulate matrices with random values $\in \{0, 1\}$ (both independent and equally likely to occur), where each column is a feature and each row is a sample, analogously to the format described in Section 2.3. We generate matrices of different sizes, starting from 10x10 to 500x500, with a step of 10 for both dimensions, resulting in 2.500 data sets. To assign the samples to the two classes, we propose two setups. In the first setup (1), we randomly (with equal probabilities of the gate and input occurrence) generate a Boolean classifier according to the core constraints proposed by Mohammadi et al., 2017 described in 2.4 and use the classifier to assign each sample to one of the two classes. In this setup, we guarantee the existence of a desired solution. In setup (2), we design classifiers according to a binomial distribution setting the upper bound of the number of gates to 10 and the number of features assigned to one gate to 5. We then randomly (with equal probabilities) choose whether each gate and input is a part of the final classifier. In this case, we do not ensure a feasible solution (fulfilling the constraints described in 2.4) exists. To each classifier, we randomly (with equal probabilities) assign features from the data set. Finally, we employ the classifiers to annotate the samples in each simulated matrix.

As the ASP solver searches through all possible solutions, we also perform a scalability study estimating our approach's run time for various data sizes. Thus, we generate additional data sets with a number of features going up to 10.000 and setting the number of samples to 50 per data set. We apply setup (1) for all scalability data sets, ensuring a feasible solution exists.

3.2.6 Breast cancer data

To evaluate the performance of the ASP-based approach on experimental data, we employed breast cancer data sets previously used by Mohammadi et al., 2017 (initially described by Farazi et al., 2011). The authors pre-processed the data sets via normalization, aggregation of similar miRNAs and removal of undesirable (e.g., premature) miRNAs as described in Mohammadi et al., 2017. The authors also provide binarization thresholds according to which the entire data set can be discretized into high (1) and low (0) expression levels. The details about the samples and miRNAs comprised by the particular data sets are presented in Table 3.1. The binarization thresholds applied to discretize the data may be found in Section S2.1.

TABLE 3.1: Breast cancer data details. Abbreviation diff. reg. stands for differentially regulated.

Dataset	Samples	Positive	Negative	miRNAs	diff. reg. miRNAs
All	178	167	11	478	57
Triple-	82	71	11	456	52
Her2+	86	75	11	438	19
ER+ Her-	32	21	11	392	18
Cell Line	17	6	11	375	59

Breast cancer subtype classification is based on the expression of estrogen (ER) and progesterone (PR) hormone receptors, as well as ERBB2 (also called HER2) gene expression. The tumour can be luminal (ERBB2-negative, ER-positive and, in the case of the luminal A subtype, PR-positive), of ERBB2+ type (also called Her2+, i.e., ERBB2-positive and at the same time ER-negative and PR-negative), or triple-negative (also called Triple-, i.e., ER-negative, PR-negative and ERBB2-negative). The data set *All* includes samples of different breast cancer subtypes (*Triple-*, *Her2+* and *ER+ Her-*) as well as 11 control samples coming from healthy tissues. The following subtype data sets *Triple-*, *Her2+* and *ER+ Her-* represent the mentioned breast cancer subtypes. The last data set (*Cell Line*) consists of cell line samples (not included in the *All* data set) and the control samples. We format the data according to the description in Section 2.3. In the case of cell classifier design, non-regulated miRNAs do not carry any information. Thus, we remove them from the data sets before optimizing the classifiers. The last two columns of Table 3.1 include numbers of miRNAs before and after removing non-relevant miRNAs from each data set.

3.3 Results

3.3.1 Simulated data studies

Performance analysis We run our approach on simulated data sets generated with setup (1) - data set annotated with a feasible solution (see Section 3.2.5), and setup (2) - a feasible solution is not ensured. In both setups, we first search for at least one feasible solution without size optimization. Then, we re-run the solver using the size optimization strategy (3) - minimizing the number of inputs followed by the number of gates. We use time-outs between 10 and 60 minutes to limit the run time.

Figure 3.3a shows that for setup (1), we can find at least one feasible solution for most problems within the maximum time limit of 60 minutes. The black squares

indicate that the problem is unsolved within the maximum run time, and the black frames that the problem is proven to be unsatisfiable, namely, a feasible solution does not exist for a given data set. Figure 3.3b shows that, in the case of size optimization, more problems (16% of all) are unsolvable in under 10 minutes compared to the previous setup. We sample 10% of unsolved problems and re-run the solver with a time-out of 5 hours. About 67.5% of the sampled problems are solvable at an average of 1 hour and 27 minutes. The remaining runs exceeded the maximal run time.

We repeat the above-described experiments with setup (2). Figure 3.3c shows that more problems are proven to be unsatisfiable with the increasing number of samples. However, above a certain number of features oscillating between 250 and 300, many problems are solved within the time limit. With the increasing number of features in the data also increases the chance that a feasible classifier can be constructed. This is not visible in Figure 3.3d, where around half of the problems reach the maximal run time.

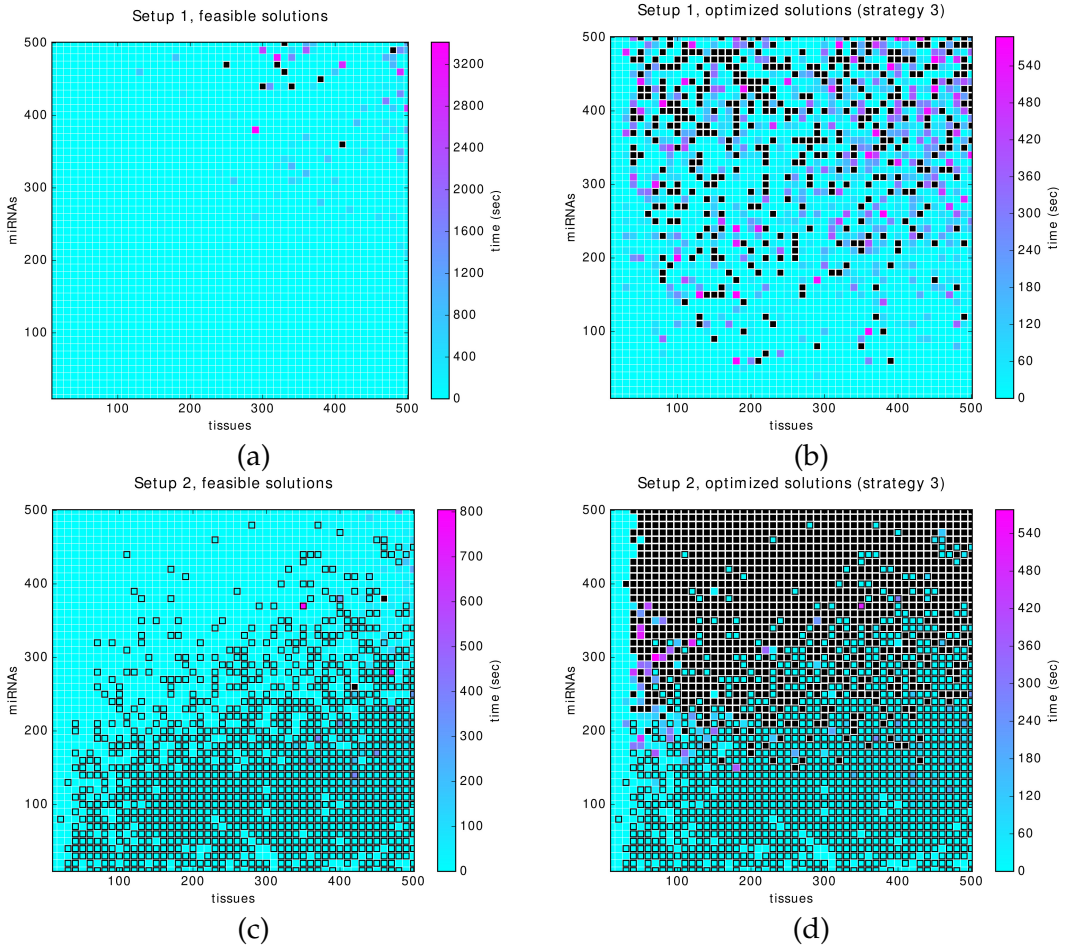


FIGURE 3.3: Performance analysis: (a) run with setup (1) and without the size optimization strategy, (b) run with setup (1) and with the size optimization strategy (3), (c) run with setup (2) and without the size optimization strategy (b) run with setup (2) and with the size optimization strategy (3). The x-axis corresponds to the number of samples and the y-axis to the number of features. The black squares indicate that the problem is unsolved within the maximum run time, and the black frames that the problem is proven to be unsatisfiable. Figures generated by HK.

Cross-validation To measure the generalization error of our classifiers, we perform 10-fold cross-validation for data sets generated with setup (1). Next, we split the data into ten pairs of disjoint training and validation fractions. Each training fraction consists of the remaining nine validation fractions. Then, we run ASP for each training fraction to find a classifier further evaluated on the validation data. We treat time-outs of 10 minutes as false predictions. In Figure 3.4a, we can distinguish two areas with visibly lower performance. First, the low number of samples in the training fraction can result in overfitting. Thus, the performance drops along the y-axis. Also, for some of the largest data sets, the run time exceeds the upper bound of 10 minutes. If no feasible classifiers are found for all ten training data sets, the generalization error is 1.0. We repeat the cross-validation with optimization strategy (3). Here, across both axes, the generalization error increases drastically, particularly with the increasing number of features.

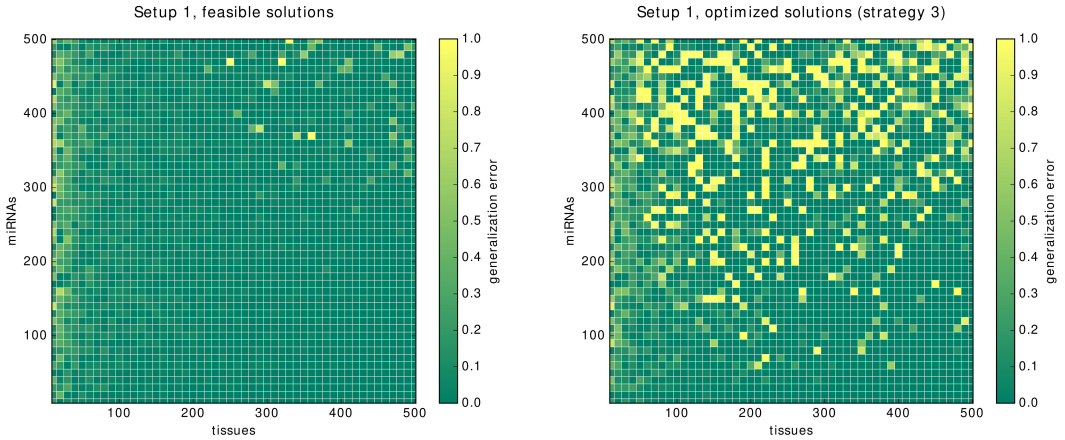


FIGURE 3.4: Cross-validation: (a) without the size optimization strategy (b) with the size optimization strategy (3). The x-axis corresponds to samples, and the y-axis to the features. Figures generated by HK.

Scalability We uniformly sample 4.500 from 10.000 data sets and run our workflow without the optimization strategy and with a time-out of 30 minutes per problem. We present the results in Figure 3.5. As expected, with the increasing number of features, the number of problems not solvable within 30 minutes increases as well. However, the average elapsed time is below 10 minutes for all data sets. Figure 3.6 shows that for different numbers of features in the data, the number of unsolved problems does not exceed 15 (one histogram bar corresponds to 400 runs).

3.3.2 Breast Cancer Case Studies

We employ the *All* dataset to asses the performance in classifying cancerous vs healthy samples. We also use each subset, as well as the cell line data, to evaluate the classification of subtype or cell line samples vs. healthy samples. For each dataset, we apply the (3) optimization mode (minimize the number of inputs followed by the number of gates) and search for a perfect classifier respecting the following *core constraints*: upper bound on the number of gates: 6, upper bound on the number of inputs: 8, and specify two types of gates based on the feasibility constraints proposed by Mohammadi et al., 2017, where an OR gate module consists of up to 2

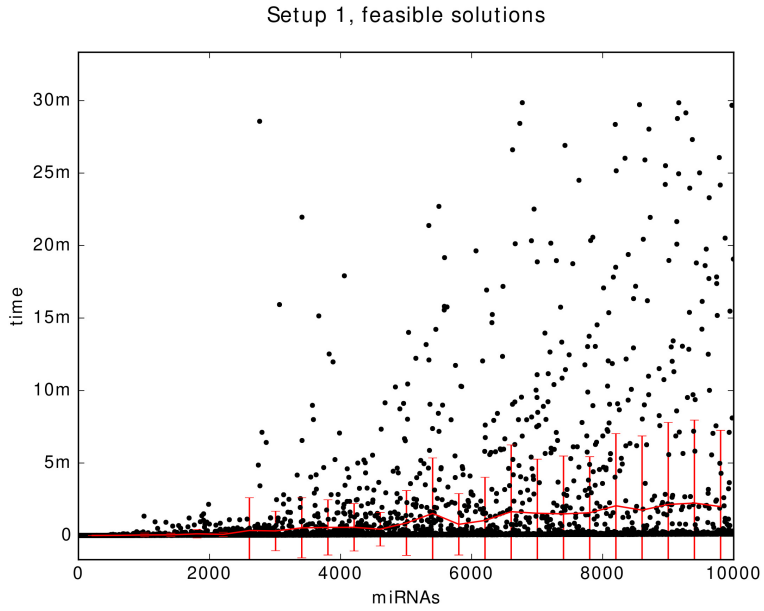


FIGURE 3.5: Scalability results: a scatter plot of the problems that are solved in under 30 minutes. Figure generated by HK.

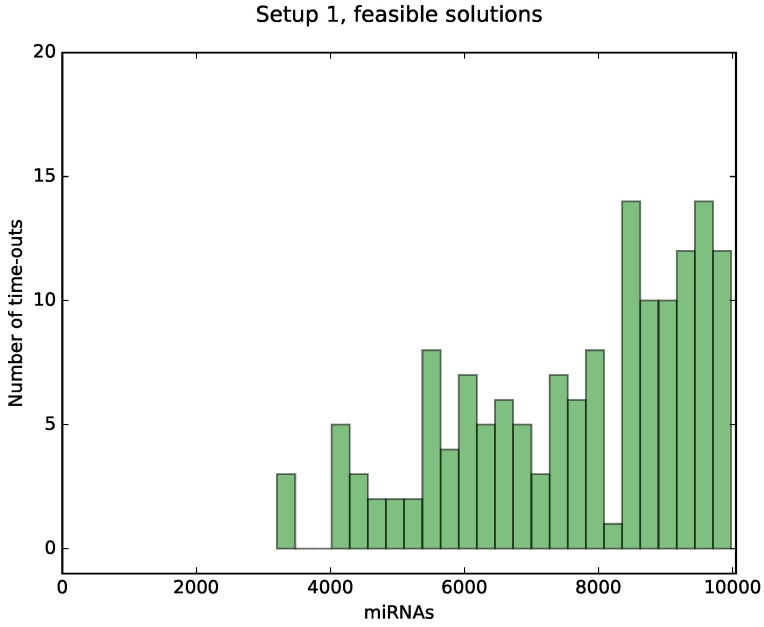


FIGURE 3.6: Scalability results: a histogram of the number of problems that are unsolvable in under 30 minutes. Each bar corresponds to 400 problems. Figure generated by HK.

miRNA inputs and a NOT gate of up to 1 miRNA input. If the search is unsuccessful, we apply the constraint relaxation procedure described in Section 3.2.2. Table 3.2 includes the returned classifiers and summarises the evaluation scores. For all data sets, the solutions were returned by the solver in seconds.

Subtype	Classifier	FNR	FPR	S_{AUC}	M_a	M_w	S_m
BC All	**(¬ miR-378)**	0.02	0.27	0.96	0.79	-0.47	0.16
Triple-	**(miR-24-1) ∧ (¬ miR-378)**	0.04	0.18	0.99	0.61	-0.10	0.25
	(¬ miR-378) ∧ (¬ miR-144)	0.04	0.18	0.98	0.81	-0.34	0.24
Her2+	**(miR-21) ∧ (¬ miR-451-DICER1)** ∧ (¬ miR-320-RNASEN)	0.00	0.09	0.99	0.87	-0.24	0.31
ER+ Her-	**(miR-21) ∧ (¬ miR-320-RNASEN)**	0.00	0.18	0.96	0.74	-0.46	0.14
	(miR-21)	0.00	0.27	1.00	0.85	0.14	0.50
Cell Line	**(¬ miR-145)**	0.00	0.00	1.00	1.66	1.33	1.50
	(¬ miR-143)	0.00	0.00	1.00	-	-	1.16
	(¬ miR-199a-2-5p)	0.00	0.00	1.00	-	-	0.96
	(¬ miR-451-DICER1)	0.00	0.00	1.00	-	-	0.93
	(¬ miR-146a)	0.00	0.00	1.00	-	-	0.55
	(miR-425)	0.00	0.00	1.00	-	-	0.32

TABLE 3.2: Evaluation of breast cancer classifiers with scores: false negative rate, false positive rate, AUC, average margin and worst margin. Best performing classifiers for a particular data set are in bold.

Breast Cancer All For the breast cancer *All* dataset, we found two classifiers presented in Figure 3.7. The first classifier consists of only one gate and one input miRNA (*miR-378*) and results in four false negative (FNR = 0.02) and three false positive (FPR = 0.27) errors. The single-miRNA classifier is easier to assemble in the wet lab. However, it is worth considering whether one input classifiers are reliable enough to tackle the real-environment diversity of cancer cells (Mohammadi et al., 2017). Results generated by MN.

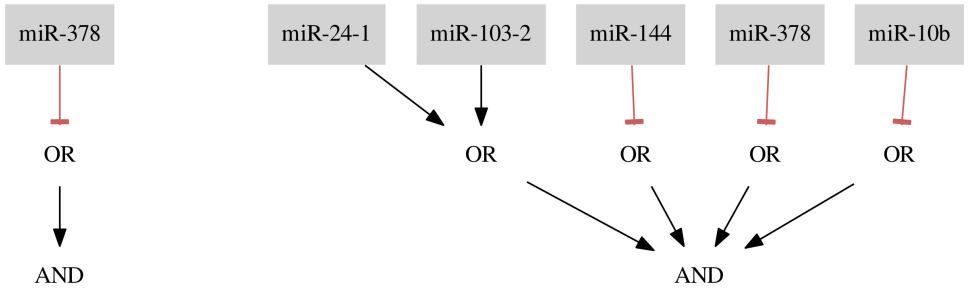


FIGURE 3.7: Classifiers for Breast Cancer All data set. Note that the classifier is in the conjunctive normal form being a conjunction (AND) of disjunctions (OR). Thus, all miRNAs are inputs to the disjunctions (OR) as negated (marked with a red line with a perpendicular bar at the end) or non-negated features (marked with an arrow). Figure generated by MN using scripts provided by HK.

As mentioned before, one type of error may be less desirable or entirely forbidden in this particular application. Hence, we re-run the optimization for the *All* data set, setting the upper bound on false positive errors to 0, restricting the misclassification of healthy cells as cancerous. In this case, we find six optimal solutions. However, all of them are isomorphic to the classifier presented in Figure 3.7. If we compare the presented classifier (miR-24-1 $\vee$ miR-103-2) $\wedge$ ($\neg$ miR-144) $\wedge$ ($\neg$ miR-378) $\wedge$ ($\neg$ miR-10b) with one of 5 remaining solutions, for example, (miR-24-1 $\vee$ miR-103-2) $\wedge$ ($\neg$ miR-378) $\wedge$ ($\neg$ miR-10b) $\wedge$ ($\neg$ miR-144) it is clearly visible that these solutions differ in the order of miRNA IDs assigned to the three NOT gates. All six isomorphic solutions differ, in fact, in the permutations of the three different IDs assigned to three NOT gates and belong to the same isomorphism class. We process the solutions to eliminate these copies as described in Section 3.2.3. Ultimately, we find only one isomorphism class. Although, in this case, the isomorphic solutions are easily distinguishable, it may happen that the computation results in a few isomorphic classes (e.g., in the case study Breast Cancer Triple- we find more than one class), and the classifiers consist of many inputs. Then, we may receive thousands of isomorphic solutions, and an automated approach to scan the solutions becomes necessary.

Both classifiers presented in Figure 3.7 share the same gate and input miRNA *miR-378*, which is down-regulated. The study of Farazi et al., 2011 describe *miR-378* as low expressed and suggests that the use of *miR-378* as a potentially down-regulated marker in a classifier is reasonable. Also, an unrelated study shows that,

e.g., *miR-144*, which occurs in the second classifier, is expected to be down-regulated in breast cancer (Pan et al., 2016).

To further assess the performance of our classifiers, we perform 3-fold cross-validation for the breast cancer *All* data set. We randomly divided the dataset into three almost equal subsets (the difference between the data sizes does not exceed 1) without taking an even distribution of positive and negative samples between subsets into account (the folds comprise between 2 and 6 negative samples). The folds consist on average of 56 positive and only 4 negative samples. To find the classifiers, we again employ the constraint relaxation procedure and find classifiers resulting in FN rate = 0.01 and FP rate = 0.56. The results show that our approach allows the training of classifiers that recognize positive samples almost perfectly. The high FP rate reflects the data's imbalance of negative and positive samples, e.g., if a test data set comprises only 2 negative samples and one of them is falsely classified as positive the FPR = 0.50. We revisit this issue in the discussion. The cross-validation resulted in 2 different classifiers: (¬ miR-144) and (¬ miR-10b) AND (¬ miR-193a-5p). Both *miR-144* and *miR-10b* appeared in the second classifier presented in Figure 3.7, and both were marked as down-regulated in different studies (Farazi et al., 2011; Pan et al., 2016). Also, *miR-193a-5p* is marked as down-regulated by Farazi et al., 2011. Despite discretization, our method recovers the signal present in the data.

Breast Cancer Triple- For the breast cancer *Triple-* dataset, we found two classifiers: the first classifier consists of two NOT gate modules, and the second of one NOT gate and one of type OR gate. Both classifiers, shown in Figure 3.8, are results of allowing at most 3 false negatives and 2 false positives (FNR = 0.04, FPR = 0.18). Here, different criteria can be taken into account to choose between the classifiers, e.g., whether some gate modules are more desired than others. This choice makes our method flexible regarding particular experiments and datasets. *miR-378* appearing in both classifiers is marked as down-regulated in a study by Farazi et al., 2011. *miR-24-1* is described as a up-regulated (Roscigno et al., 2017) and, as mentioned before, *miR-144* is described as down-regulated in breast cancer by unrelated studies (Pan et al., 2016).

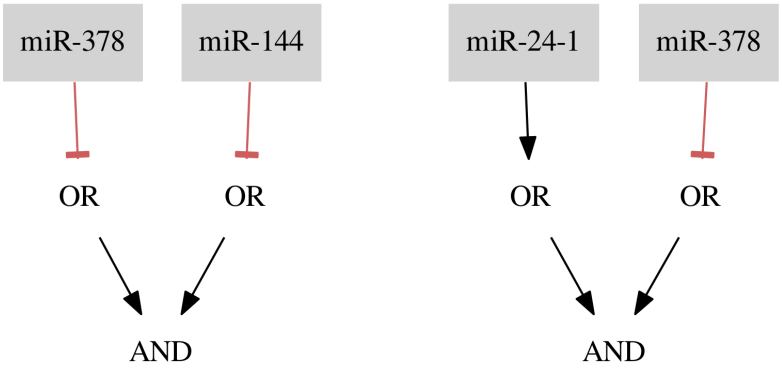


FIGURE 3.8: Classifiers for the breast cancer *Triple-* data set. Figure generated by MN using scripts provided by HK.

Breast Cancer Her2+ The resulting classifier for the dataset *Her2+* (shown in Figure 3.9) consists of three inputs and three gates: one OR gate with one input and two

NOT gates (FNR = 0.00, FPR = 0.09). *miR-451-DICER1* and *miR-320-RNASEN* are marked as down-regulated and *miR-21* as up-regulated in a study by Farazi et al., 2011.

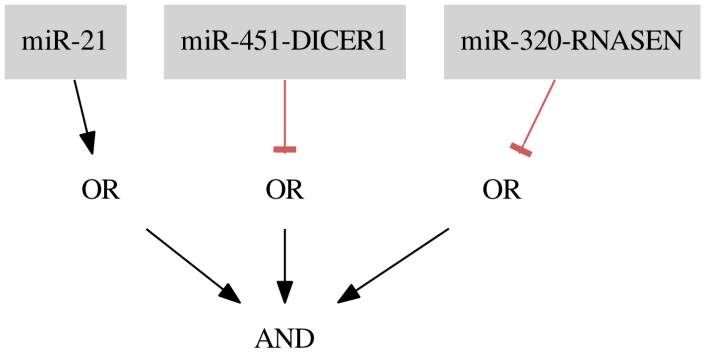


FIGURE 3.9: Classifier for the breast cancer *Her2+* data separating samples with one false positive error. Figure generated by MN using scripts provided by HK.

Breast Cancer ER+ Her- For the *ER+ Her-* dataset we found one classifier shown in Figure 3.10A on the left side. The classifier consists of two gates of both types and two inputs resulting in a classification with two false positive errors (FNR = 0.00, FPR = 0.18). Here, we present an additional classifier with only one input forming an OR gate resulting in classification with three false positive errors (FNR = 0.00, FPR = 0.27) shown in Figure 3.10A on the right side. In this case, we are able to obtain a shorter classifier relaxing the bounds on the number of false positives by only one additional error. It is worth considering whether the one misclassified sample is reliable or if we could neglect it and build a simpler classifier. Also, in case one of the miRNAs cannot be included in the classifier or the classifier consists of too many inputs, it may be worth further increasing the bounds on errors. Note that the first classifier consists of only one gate, and the same gate is a part of the second classifier. *miR-21* is marked as up-regulated and *miR-320-RNASEN* as down-regulated by Farazi et al., 2011.

Breast Cancer Cell Line For the *Cell Line* dataset, the optimization results in six perfect classifiers that consist of only one gate and one input each, where five of them consist of single NOT gate modules and only one consists of a single OR gate module. These classifiers distinguish cancerous from healthy samples based merely on the expression level of one miRNA (FN rate = 0.00, FP rate = 0.00). As an example, we present a classifier with a negative input of miRNA *mir-145* (see Figure 3.10B). The classifier predicts every sample to be cancerous if *mir-145* is at a low expression level. All other samples are predicted to be healthy. *mir-145* is also marked as down-regulated in a study by Farazi et al., 2011. The six resulting classifiers are

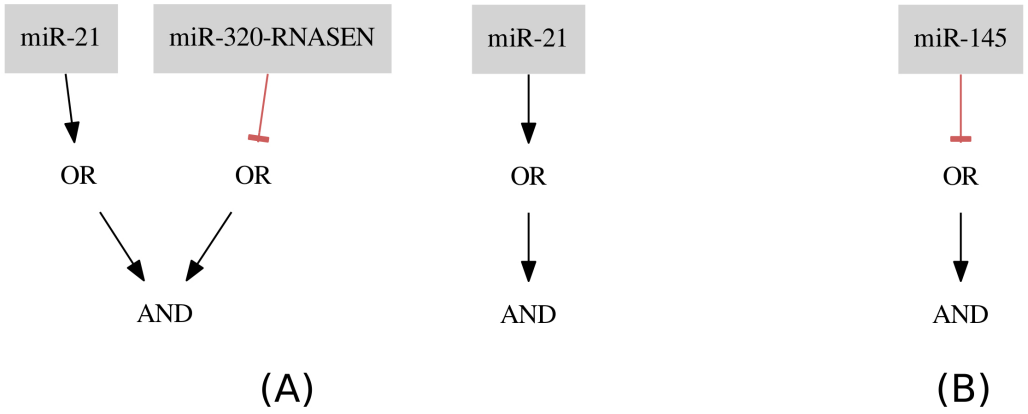


FIGURE 3.10: Results for the breast cancer (A) *ER+ Her-* and (B) *Cell Line* data sets. Figure generated by MN using scripts provided by HK.

presented in Table 2. In this case, the miRNAs can also be combined into one classifier to increase the robustness of classifiers. In the next section, we discuss additional optimality criteria for choosing between several perfect classifiers.

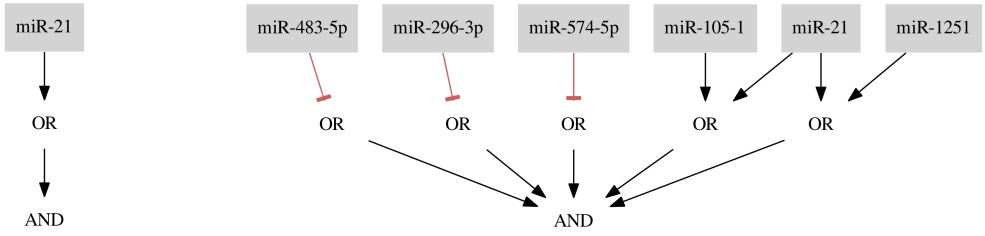


FIGURE 3.11: Comparison of the shortest ASP-generated classifier (left) with the classifier proposed by Mohammadi et al. (right) for breast cancer ER+ Her-. Figure generated by MN using scripts provided by HK.

Approach comparison In this section, we compare the performance of classifiers optimized with the ASP-based approach and the state-of-the-art method proposed by Mohammadi et al., 2017. We focus on the primary algorithm proposed by the authors, namely, the specific search (described further in Section 2.6). In their work, the authors compare the specific and general search approaches and conclude that the specific search performs with higher accuracy in comparison to the general search according to the metrics described in 2.6. We consider only pruned circuit designs, i.e., the circuits post-processed by the authors to decrease the overall number of inputs (without substantial influence on the accuracy of classifiers).

The authors assess the classifier's performance in two setups. First, they utilize entire data sets in training and report training performance for all the breast cancer data sets. Then, the authors perform 3-fold cross-validation to estimate the generalization error of the classifiers. We follow this strategy and report training accuracy for all data sets and the performance measured in 3-fold cross-validation for the breast cancer *All* data set. We report the performance of the circuits in two settings: (1) In the binary setting, we compute FNR and FPR for each circuit. (2) In the non-binary setting, we report S_{AUC}, S_m, as well as M_a (average margin), M_w (worst margin), proposed by Mohammadi et al., 2017 and described in Section 2.6. Table 3.3 includes the performance obtained in training by both methods. We compute all the scores keeping the same biochemical parameter sets and binarization thresholds for each dataset as proposed by Mohammadi et al., 2017 and compare classifiers presented by the authors (Table S3.1) with ours (Table 3.2). If we receive more than one equally well-performing classifier, we follow the strategy proposed by Mohammadi et al., 2017 and select the one with the highest S_{AUC} and S_m for comparison. Note that Mohammadi et al., 2017 optimize the classifiers in a different modelling framework, while our classifiers are optimized solely on the binary data set. Nevertheless, the central goal of both approaches is to find minimal and accurate classifiers. Also, note that in terms of the binary classification scores, our classifiers are the optimal solutions for each data set with regard to the binary setting, i.e., a better solution does not exist for a particular data set.

Data set	Method	S_{AUC}	M_a	M_w	S_m	FPR	FNR
All	SynNet	1.00	0.78	0.01	0.40	0.00	0.63
	ASP	0.96	0.79	-0.47	0.16	0.27	0.02
Triple-	SynNet	1.00	0.94	0.08	0.51	0.00	0.10
	ASP	0.98	0.81	-0.34	0.24	0.18	0.04
Her2+	SynNet	1.00	0.87	0.18	0.53	0.27	0.00
	ASP	0.99	0.87	-0.24	0.31	0.09	0.04
ER+ Her-	SynNet	1.00	0.92	0.37	0.65	0.27	0.13
	ASP	0.96	0.74	-0.46	0.14	0.18	0.00
Cell Line	SynNet	1.00	1.90	1.51	1.71	0.00	0.00
	ASP	1.00	1.66	1.33	1.50	0.00	0.00

TABLE 3.3: Evaluation scores for SynNet and our method: $\overline{m}$ (average margin), w (worst margin), S_{AUC}, FNR and FPR for each circuit. Corresponding circuits are presented in Table S3.1. MN generated results for ASP. Results for SynNet were partially obtained from Mohammadi et al., 2017. Additionally, MN computed FPRs and FNRs for the SynNet classifiers.

Although we optimized our classifiers using different optimality criteria, our circuits perform similarly well regarding the S_{AUC} score, which is the primary score considered by Mohammadi et al., 2017. According to the second score, S_m, the ASP-optimized classifiers perform worse in comparison to SynNet. For all data sets besides *Cell Line*, the worst margin is negative, substantially influencing the S_m score. The worst margin describes the distance between the closest data points in the two classes and thus concerns only two samples. Nevertheless, the negative worst margin indicates that at least one sample is misclassified, and the effectiveness of the

therapy decreases. This is not the case in Mohammadi et al., 2017 as the authors optimize the AUC and the margin-based scores directly. In contrast, we rely exclusively on the absolute number of errors in the binary setting. Thus, one misclassified sample likely results in the low worst margin.

Strikingly, although the methods perform comparably regarding the non-binary metrics, there is a large discrepancy between the FPRs and FNRs reported for both methods. The classifiers optimized with the specific search perform worse for most data sets according to the binary metrics, whereas according to the objective criteria proposed by Mohammadi et al., 2017, the circuits perform with AUC=1.00. In particular, FNR for the *All* data set is substantially higher, resulting in more than 60% misclassified samples belonging to the positive class. We address this problem in the following section.

We employed slightly different constraints than in Mohammadi et al., 2017 to optimize the logic classifiers (maximum of 8 inputs, 10 in Mohammadi et al., 2017). Here, extending the size of the classifier would enable obtaining better results in terms of ASP-based optimization, as ASP allows enumerating all globally optimal solutions for particular constraints in the binary setting. Thus, we relaxed size constraints for all the data sets except for the *Cell Line* (for which we received perfect classifiers). This enabled improving the performance of classifiers for *All* (FPR = 0.18, FNR = 0.02), *Triple-* (FPR = 0.09, FNR = 0.04) and *ER+ Her-* (FPR = 0.09, FNR = 0.00) datasets. In all cases, the size of classifiers increased due to additional inputs allowed in the OR gate modules.

The results for the 3-fold cross-validation reported by Mohammadi et al., 2017 (S_{AUC} = 0.99, Sm = 0.31) and ours (S_{AUC} = 0.93, Sm = 0.24) show that our method separated the unseen samples with slightly worse, but similar performance according to the non-binary metrics. Note that the samples are divided into random subsets by us and by Mohammadi et al., 2017 independently. Mohammadi et al., 2017 do not publish the data and the classifiers optimized in the cross-validation. Thus, we do not report FPR and FNR. However, the similarity of performance suggests that the different data splits did not substantially affect the results. We revisit this issue in the discussion.

In terms of size, we find shorter classifiers for all data sets and improve their performance according to the binary metrics. The larger difference in size is visible for the *ER+ Her-* data set for which we found a classifier consisting of single miRNA, whereas Mohammadi et al., 2017 proposed a classifier comprising seven features (depicted in Figure 3.11).

3.4 Boolean approximation of the circuit

In this section, we address the issue related to the Boolean representation as an approximation of a circuit's biochemical model, visible in the discrepancy between the performance metrics employed in the previous section. The S_{AUC}=1.00 obtained by all classifiers presented by Mohammadi et al., 2017 indicates the perfect classification for all the data sets based on the circuit output concentration. However, the SynNet classifiers misclassify several samples when evaluated on the binarized data. Here, we present a few examples of disparities between the binary representation and the circuit's models.

In Figure 3.12, we present a plot of the output concentrations for the circuit optimized by Mohammadi et al., 2017 for the breast cancer All data set. We compute the concentrations based on the biochemical model and parameters assigned to each

data set. The exact circuit output concentrations in ascending order computed for 20 samples can be found in Table S3.2. The circuit output separates all the samples in the data set into two groups. The outputs for the samples closest to each other (negative sample 7 and positive sample 178) are separated by approximately 2.5 mol/cell, leaving little margin for error (captured by the worst margin $M_w = 0.01$). Here, one could consider merging S_m and S_{AUC} into a weighted score to increase the robustness of the classifiers.

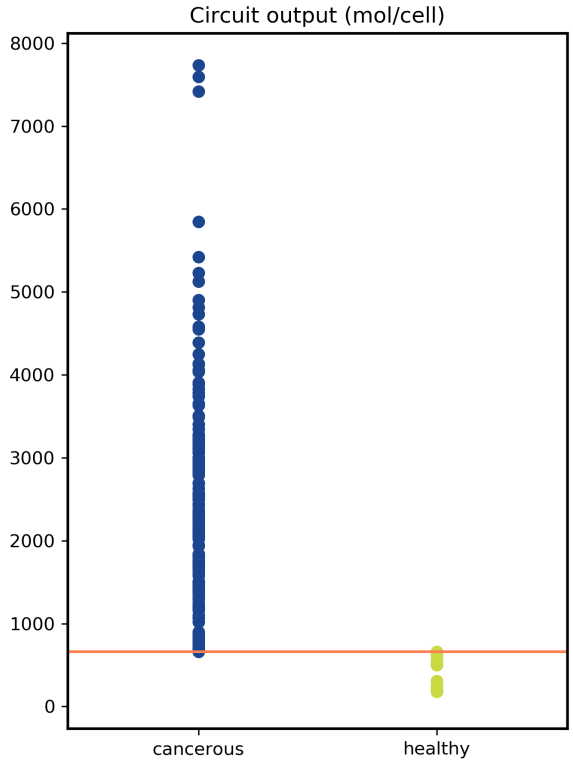


FIGURE 3.12: Circuit output concentrations in two groups calculated for the breast cancer All data set. A feasible threshold separating the samples is plotted as a light red line. Note that the centre of the point represents its position (see Table S3.2 for exact values of the output concentration).

Evaluation of the SynNet Boolean classifier on the binary *All* data set results in 105 false predictions (FNR=0.63), indicating a discrepancy between the model and the logic representation of the circuit's architecture. We can observe the same scenario for the following 3 data sets: Triple-, Her2+ and ER+ Her- (Table 3.3). We analyze two types of errors occurring for these data sets (false negatives and false positives) and evaluate which parts of the circuits are involved in the classification failure. The details are presented in the Appendix S3.3-S3.4. In both cases, certain miRNA concentrations exceed the binarization thresholds and therefore are misclassified in the binary setting. As previously mentioned, the Hill functions are well-approximated by a Boolean term only in the case of very high or very low concentration values. However, such an approximation does not account for values that are close to the threshold. Therefore, the comparison of classifiers optimized based on

real-valued miRNA concentrations and based on binarized values is not trivial. The margin could be considered within the optimality criteria to increase the robustness of the Boolean representation of the circuit, especially in extreme cases such as the breast cancer *All* data set.

3.5 Discussion

In this chapter, we introduced an ASP-based approach to cell classifier design. We developed and implemented a workflow finding globally optimal Boolean classifiers based on binary data, enabling the enumeration of all optimal solutions in a short time. We employ simulated data sets to measure the overall performance of our method, including the generalization error and scalability. Further, we apply our approach to cancer data and compare it with the state-of-the-art method.

The simulated data studies have shown that for most problems in which feasible classifiers exist, the solver returns solutions in below ten minutes of elapsed time for hundreds of samples and features in the data set. The ASP solver can easily retrieve a classifier created in the simulation process. If the simulation setup does not ensure a feasible solution, our workflow returns solutions for most of the problems in an extended run-time, if they exist. Otherwise, the solver proves numerous problems unsatisfiable in a short time. Adding the size optimization strategy seems to increase the run time substantially. However, many problems may potentially be solved within extended time limits.

According to the performed cross-validation, the optimized classifiers generalize well to unseen data, particularly if the size optimization strategy does not limit the classifier's structure. However, all the samples in the data set have been annotated with a simulated feasible solution (see section 3.2.5). This can be interpreted as information leakage between the training and validation fractions of the data set. To properly estimate the generalization error in a more realistic scenario, the training, validation and test data should reflect the use-case scenario. However, information not related to creating a suitable evaluation environment shared between the data sets should be minimized to avoid distorted generalization error estimation (Whalen et al., 2022). Employing the size optimization strategy again increases the run-time of the workflow, and as we treat time-outs as misclassification, the generalization error is also very high. This effect could be potentially reduced by extending the run time limits. We revisit these issues in Chapter 5, which focuses on refined strategies to simulate a comprehensive evaluation environment and mitigate the influence of information leakage to enable properly estimating the generalization error of classifiers.

The scalability study shows that the proposed workflow is flexible regarding the number of miRNAs in the data set. However, the influence of the increasing number of samples could be further investigated, as the scalability study was performed for only 50 samples. We address this issue in Chapter 5 by performing an extensive scalability test employing data sets comprising different numbers of samples and features. Note that in many cases, the ASP encoding influences the scalability of the solving process (Falkner et al., 2018). In our case, most of the simulated problems have been solved in a matter of minutes. Thus, the proposed encoding seems efficient, although it could be potentially improved by, e.g., preventing symmetric solutions in the solving process by embedding proper constraints into the ASP program (Falkner et al., 2018).

To assess the performance of our approach in a real-world environment, we employed breast cancer data sets. Our classifiers correctly recognize most cancerous samples for all data sets according to the reported False Negative Rate. The False Positive Rate is lower, potentially indicating the higher treatment's toxicity. However, the data sets employed in the study are heavily imbalanced. In particular, the negative class is substantially underrepresented, providing a small number of negative examples in training and potentially influencing the classifier's performance (Whalen et al., 2022). Thus, employing alternative data sets or balancing strategies that mitigate this issue is recommended. We revisit this issue in Chapter 4 and apply balanced accuracy that averages the sensitivity and specificity of classifiers as the objective function and performance measure to reduce the influence of class imbalance on the training and evaluation of classifiers. Also, as the approach is tested solely on breast cancer data, evaluating the approach on different cancer data sets would be beneficial. In Chapter 5, we present multiple case studies that employ different cancer data sets, in particular, liver and lung cancer data. Although we tailor our implementation to a particular application, the workflow can be applied to any problem that requires finding Boolean classifiers for binary data. The flexibility of our encoding could be further investigated by, e.g., applying the workflow to alternative data sets beyond the cancer context.

We compare the performance of our approach with the state-of-the-art method proposed by Mohammadi et al., 2017 by evaluating the classifiers in two scenarios, using binary and non-binary metrics. Although our classifiers are optimized based solely on the Boolean topology and the overall number of errors, they perform notably well in the non-binary setting. The lower performance is particularly visible in the margin separating the closest negative and positive samples with regard to the output concentration. However, in the binary context, our classifiers outperform the designs optimized by Mohammadi et al., 2017. The performed comparison focuses on the training performance of classifiers. This does not reflect well the real-world application of cancer classifiers. We address this issue in the following chapters by estimating the generalization error of the ASP-based approach in different scenarios.

The approach comparison uncovers a considerable discrepancy between the binary and non-binary metrics in the case of SynNet-optimized classifiers. We further investigate this issue by stepwise evaluating particular samples and classifier topologies. We focus on two aspects of this issue. First, Mohammadi et al., 2017 use the AUC score as the primary measure of the classifier's performance. However, the margin score seems to be equally important. Even though the samples are separated into two classes by AUC, the margin between the output concentrations in the cancerous and non-cancerous cells is very narrow. Thus, incorporating the margin score into the primary objective function could be advantageous. Second, although the Boolean term can approximate the model, it has to follow the underlying assumptions. Thus, in the case of the near-threshold miRNA concentrations, the robustness of the Boolean representation seems to be decreased. In the approach proposed by Mohammadi et al., 2017, the binarization threshold corresponds to a set of biochemical parameters used in the optimization process and describes the model's behaviour below and above the binary threshold. However, when fully translating the real-valued model into a binary classification problem, a margin of error around the threshold should be considered to increase the robustness of the Boolean representation. In Chapters 4 and 5, we focus on increasing the robustness of Boolean classifiers employing different classifier topologies and refining the training scheme to increase the generalization error.

Chapter 4

From single- to multi-circuit classifiers

Contribution note

The greater part of the work presented in this chapter has been published in the Proceedings of the 17th International Conference on Computational Methods in Systems Biology, Lecture Notes in Computer Science by Springer (Nowicka and Siebert, 2019). The employed data and code are stored in a GitHub repository at: `https://github.com/MelaniaNowicka/RAccoon`.

4.1 Introduction

The studies by Mohammadi et al., 2017 and Becker et al., 2018 (see Chapter 3) have shown the potential of single-circuit classifiers to perform with high accuracy in the task of cancer cell classification. However, as described in Chapter 1, the heterogeneity of cancer cells within a tumour usually requires very robust and versatile therapies to encompass the variety of molecular patterns, as well as to avoid tumour escape mechanisms. Moreover, the dynamic environment that classifiers enter frequently generates ambiguous perturbation events, decreasing therapy effectiveness (Dagogo-Jack and Shaw, 2018). A further limitation lies in the data gathered to design the circuits. The performance of a classifier strongly relies on the quantity and quality of data employed for training (Yang et al., 2006; Haixiang et al., 2017). Here, the coverage of patterns occurring in the tumour is limited by the finite number of acquired samples, sample purity, data pre-processing, and experimental artefacts. Hence, the data represents an incomplete description of the tumorous and non-tumorous environment. For the therapy to be effective, the classifiers must be resistant to noise and generalize to novel examples inadequately described by the data. Here, more complex therapies covering underrepresented and heterogeneous patterns could compensate for the insufficient environment depiction. Mohammadi et al., 2017 briefly mention a possibility of developing multi-circuit classifiers that could be potentially advantageous in countering the tumour escape mechanisms, mirroring other combined therapies aiming at multiple targets (Rubinfeld et al., 2006; Bozic et al., 2013; Xu et al., 2015).

Furthermore, both the complexity in the number of inputs and the allowed combinations of gates in a single-circuit classifier are limited by the lab assembly constraints. The classifier's complexity potentially rises with the variety of patterns hidden within the data, finally becoming not feasible to build (Mohammadi et al.,

2017). The architecture of single-circuit classifiers can also be prone to lower sensitivity in the case of functions consisting of many gates and inputs. Here, the Conjunctive Normal Form underlying the classifier's design enforces that all the gates must output *"cancerous"* to classify a sample as diseased. Thus, a single gate has a significant influence on the overall decision of the classifier, which, taking the heterogeneity into account, can decrease the classifier's ability to recognize cancerous cells correctly. On the other hand, a single-miRNA or a single-gate classifier can result in decreased specificity, resulting in misclassifying healthy cells. Balancing those effects can increase the effectiveness of the therapy and lower the risk of harming healthy tissues.

To address the issues mentioned above, the principles of ensemble learning can provide a more flexible solution and potentially increase the accuracy of classifiers. Ensemble learning employs multiple alternative models with possibly lower predictive performance that can yield better results collectively (Opitz and Maclin, 1999a; Polikar, 2006; Rokach, 2010). An example of such an approach is a random forest classifier, where a group of decision trees collectively determine the output of a forest of trees. A common strategy combining the individual tree outputs to decide on the final outcome is the majority vote. Here, the classifier's decision is equal to the output of the majority of trees.

Regarding the cell classifier problem, we can design a set of different single-circuit classifiers that perform classification in an integrated manner and are of lower complexity in terms of lab assembly. In other words, we design a set of simple Boolean functions that satisfy the assembly constraints. A theoretical design of a so-called distributed classifier based on synthetic gene circuits was presented by Didovyk et al., 2015. Here, the distributed classifier is a population of genetically engineered microbial cells containing simple synthetic biosensors sensitive to specific chemicals. The population forms a single complex classifier trained to find biological patterns according to a threshold function (Didovyk et al., 2015; Kanakov et al., 2015). The classifier is optimized by training a biosensor population on the available data, similarly to machine learning algorithms, i.e., by presenting learning examples and successively removing low-performance circuits. While this work considers only a specific scenario of bacterial cell cultures classifying chemical signals, it highlights the potential of multi-circuit biological devices.

In this chapter, we re-define the distributed classifier proposed by Didovyk et al., 2015 in terms of the cell classifier design. We introduce a distributed classifier (DC) as a set of single-circuit classifiers (multi-circuit classifier) that decide collectively whether a cell is cancerous based on a threshold function. Biologically, the threshold may correspond to a specific concentration of the drug that allows for treating the cells or fluorescent marker allowing to classify the cell type (Didovyk et al., 2015; Mohammadi et al., 2017; Miki et al., 2015). Due to an ample search space of feasible classifiers and a novel topology, in particular, the incorporation of the thresholded function resulting in considerably more combinations and more difficult ASP implementation, we apply a heuristic approach to design and optimize DCs, namely, a genetic algorithm. Evolutionary algorithms were successfully applied to various biological questions (Manning et al., 2013), e.g., the design of synthetic networks and, in particular, the design of single-circuit classifiers (Smith et al., 2017; Mohammadi et al., 2017). Since GAs are problem-independent and particularly flexible in design, the algorithm can be efficiently adapted to the distributed classifier problem.

The chapter illustrates the potential of distributed classifiers in the application to cancer cell classification. The following sections contain the definition of a distributed classifier and the description of the algorithm's architecture. Further, we

present case studies performed on real-world breast cancer data, compare the results with a single-circuit design method proposed in Chapter 3 (Becker et al., 2018) and discuss the classifier's performance.

4.2 Methodology

4.2.1 Distributed classifiers

A **Distributed Classifier (DC)** is a finite set $DC = \{f_1, ..., f_c\}$, where each $f_i : S \longrightarrow \{0, 1\}$ is a unique single-circuit classifier (rule), S is a finite set of samples as introduced in section 2.3 and $c \in \mathbb{N}$ is a size of a distributed classifier, i.e., the number of single-circuit classifiers in a DC. The size c is restricted by an upper bound $c_{max} \in \mathbb{N}$, $c \leq c_{max}$. DCs classify cells as non-cancerous (0) or cancerous (1) according to a threshold function specifying the classifier's output $O(DC, s) : S \longrightarrow \{0, 1\}$:

$$O(DC, s) = \begin{cases} 0\,, & \sum_{i=1}^{c} f_i(s) < \theta \\ 1\,, & \sum_{i=1}^{c} f_i(s) \geq \theta \end{cases}\,, \tag{4.1}$$

where $s \in S$ is a sample, $\theta = \lfloor \alpha \cdot c \rceil$ is a decision threshold (rounded half up), and α is the ratio that allows calculating the decision threshold based on the DC's size. The α value corresponds to the minimal fraction of rules in the DC that must output 1 (cancerous) for the DC to classify a sample as cancerous. Otherwise, the sample is recognized as healthy. An example of a DC is a set consisting of three rules and four miRNAs as inputs: (1) miR-a AND miR-b, (2) NOT miR-c, (3) miR-d. Here, (i) for low α ($\theta = 1$), at least one rule must output 1 for the DC to output 1, (ii) for intermediate α ($\theta = 2$), at least two of the rules must output 1 as shown in Figure 4.1a, (iii) for high α ($\theta = 3$) all of the rules must output 1. In the case of the low α, the function becomes a disjunction of the rules. In the second, the DC decides according to a verdict of the majority. In the last case, the function becomes a conjunction of rules being similar to a large single-circuit classifier. Thus, the classifier's output may substantially differ for varying values of α, influencing the sensitivity and specificity of the classifiers as further discussed in section 4.3.3. If the threshold is not reached, the classifier's output is 0 resulting in repression of the drug release (see Figure 4.1b).

Further restrictions on the general design of DCs were introduced to adapt the circuit's feasibility in terms of lab assembly. Motivated by section 2.4, a rule must satisfy the architecture constraints imposed on single-circuit classifiers. The number of miRNAs in a rule is restricted to 2 inputs connected with an AND operator to simplify the rule design. Further, to reduce the overall complexity of DC's architecture and compare the performance of both single-circuit and multi-circuit designs, the number of rules is restricted to 5. Hence, the maximal number of inputs in a DC is limited to 10 as proposed for SC classifiers by Mohammadi et al., 2017. We apply the above-described constraints to case studies presented further in this chapter. We assume that each rule in the DC must be unique. Thus, each rule has a single vote regarding cell classification. Also, two identical miRNA IDs cannot occur in one rule, i.e., a trivial false function ($a \wedge \neg a$) is not allowed.

4.2.2 Genetic algorithms

In our work, we employ a genetic algorithm to optimize multi-circuit classifiers. Genetic Algorithms (GAs) belong to the broader class of evolutionary algorithms (Mitchell, 1996; McCall, 2005). As in the case of EAs, the GA's core architecture is

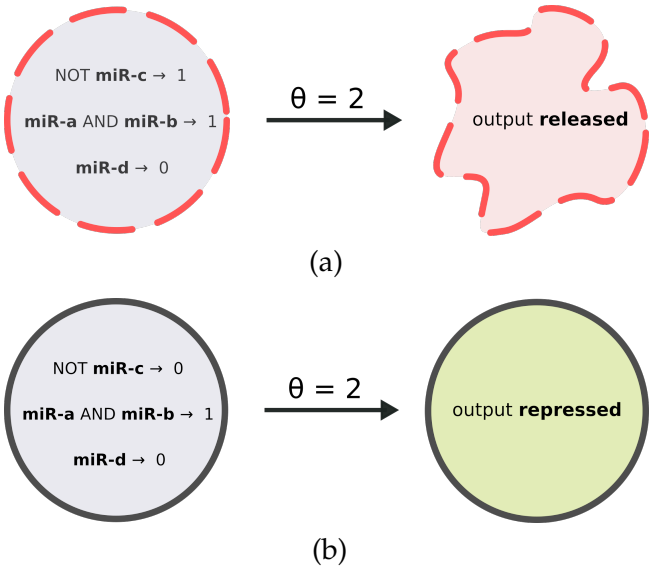


FIGURE 4.1: Exemplary distributed classifier's behaviour for the threshold $\theta = 2$. (a) Two of SC classifiers output 1 (cancerous), and the drug is released. (b) Only one of SC classifiers outputs 1, and drug release is repressed.

not designed for a particular problem providing a flexible algorithmic toolbox of operators. Thus, GAs may be applied to an array of different search and optimization problems (Manning et al., 2013; Katoch et al., 2021). Here, finding an optimal solution is not guaranteed due to the heuristic character of the algorithm. However, GAs often perform well in many different areas of research and return solutions at reasonable computational cost (McCall, 2005; Manning et al., 2013; Katoch et al., 2021). The general architecture of a GA is depicted in Figure 4.2.

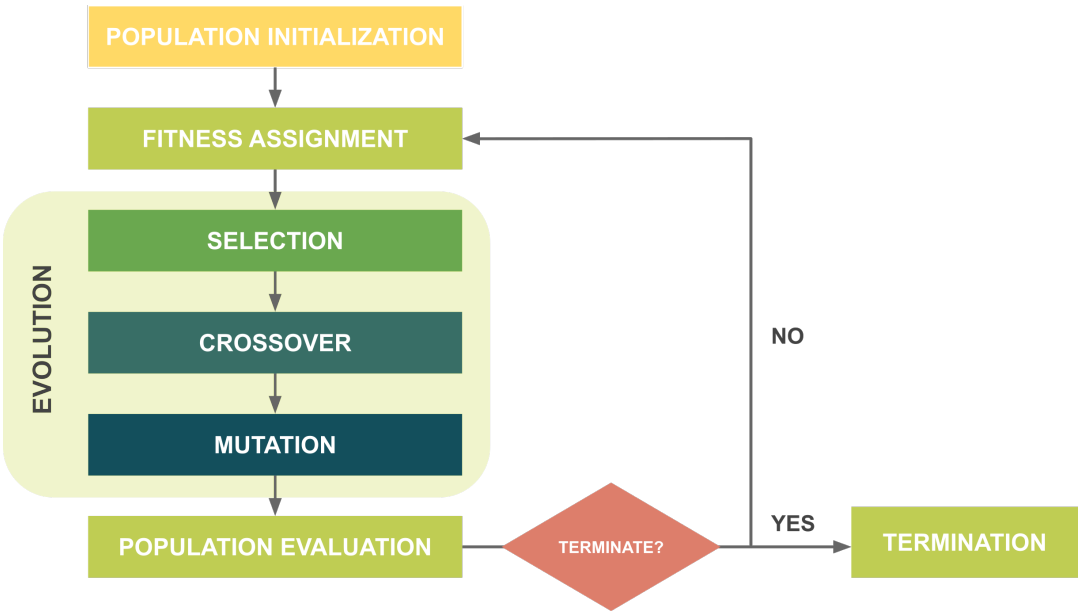


FIGURE 4.2: General architecture of a Genetic Algorithm.

The algorithm begins with initializing the first population of feasible solutions, so-called individuals, in terms of GAs. Usually, the initial solutions are generated randomly. However, the first population can also be pre-optimized to seed the algorithm's starting point potentially closer to the optimum (Kazimipour et al., 2014). Each solution consists of properties specifying the genotype that translates to the individual's phenotype. In a simple exemplary scenario, we can consider an individual represented by five different features (genes) that are either on (1) or off (0). Such a solution may be encoded as an array of 1 and 0 (genotype), e.g., [1, 0, 1, 0, 0] where features 1 and 3 are on, and features 2, 4 and 5 are off. A particular combination of feature states allows the best adaptation (fitness) to particular conditions. The individual's fitness can be described by the function $F(x_i)$, where x_i is the i-th individual in a population P of size n, $n \in \mathbb{N}$. As the optimization goal, the objective function $F(x_i)$ can be maximized or minimized.

After the population initialization, the fitness of each individual in the population is assigned, starting the first iteration of the GA. The population usually evolves in three steps: selection, crossover and mutation of individuals. Selection is a process in which a fraction of the existing population is chosen (selected) to reproduce. The parents are selected based on their fitness, i.e., better-adapted individuals are usually more likely to be chosen. Several selection operators can be applied, e.g. roulette wheel selection, rank selection or tournament selection (McCall, 2005; Katoch et al., 2021). Selected parents undergo reproduction, and a new population of offspring (new generation) is created. During reproduction, the parent's features can, with a certain probability, undergo recombination in the crossover process, exchanging parts of their genotypes. A variety of crossover operators were proposed, e.g., k-point crossover or uniform crossover (McCall, 2005; Katoch et al., 2021). After crossover, the newly generated population of offspring undergoes the mutation process. Here, each of the individuals mutates, i.e., a part of the genotype is altered, with a given probability (mutation probability) (McCall, 2005; Katoch et al., 2021). In the previously mentioned example, a mutation of an individual can be a switch on/off of one of the genes, e.g., [1, 0, 1, 0, 0] $\longrightarrow$ [0, 0, 1, 0, 0] (gene at the first position is switched off). Crossover and mutation enable maintaining diversity in a population, ensuring that the population does not prematurely converge. After mutation, the new population is evaluated according to the optimality criteria, and the next iteration of GA begins unless a termination criterion is met. The termination criterion can be specified by, e.g., a fixed number of GA iterations after which the algorithm stops, measuring the population convergence or using more complex methods (Liu et al., 2018). After the termination, the best solutions are returned.

4.2.3 Proposed architecture

In this section, we provide a general description of the algorithm's architecture, including the encoding of classifiers (solutions) and employed operators and parameters. The core architecture of the proposed GA is summarized by Algorithm 1. The detailed algorithms regarding particular components, e.g., employed operators, may be found in section S4.

As an input, the algorithm takes a discrete dataset D formatted as presented in section 2.3, a set of parameters describing the GA's run: t_{iter} - number of GA iterations until termination, p_{size} - population size, c_{prob} - crossover probability, m_{prob} - mutation probability, t_{size} - tournament size, and the design of DCs: c_{max} - maximal size of a DC, α - the decision threshold ratio. The algorithm starts with a random generation of an initial population of p_{size} distributed classifiers (Algo. 1, line 1). Next,

the fitness of each DC in the population is assigned according to the fitness function, and a list of best solutions found over all iterations of the algorithm DC_{best} is created (Algo. 1, 2). After the generation of the first population, the algorithm starts with the initial iteration. First, p_{size} individuals are selected in so-called tournaments as potential parents to be recombined, i.e., exchange genes in the crossover process. (Algo. 1, 4-7). Next, the crossover occurs with the probability c_{prob} (Algo. 1, 8-13). Crossover allows generating new solutions based on previously selected individuals. Here, a child classifier may be created by copying rules from parent classifiers and randomly choosing which parent the next rule is duplicated from. As classifier sizes may differ, we propose two uniform recombination strategies described further in section 4.2.6. Next, individuals in the new population mutate with the probability m_{prob} (Algo. 1, 14). At the end of each iteration, the list of best solutions DC_{best} is updated (Algo. 1, 15). All the described steps in a generation are repeated t_{iter} times (Algo. 1, 3-16). After reaching the t_{iter}-th iteration, the algorithm returns the best solutions found over all the iterations. In the case of single-circuit classifiers, besides the balanced accuracy, the complexity of a solution is also taken into account (Mohammadi et al., 2017; Becker et al., 2018). Hence, if multiple DCs have the same fitness, we choose the DCs consisting of the lowest number of rules as the best final candidates.

Algorithm 1: A genetic algorithm for designing DCs.

Data: dataset D
Parameters: number of iterations t_{iter}, population size p_{size}, crossover probability c_{prob}, mutation probability m_{prob}, tournament size t_{size}, maximal size of DC c_{max}, threshold ratio α
Output: DC_{best}

1 $Population \longleftarrow$ InitializePopulation(D, p_{size}, c_{max})
2 $DC_{best} \longleftarrow$ Evaluate($Population$, D, α)
3 **for** $i = 0$ **to** t_{iter} **do**
4 **for** $i = 0$ **to** $p_{size}/2$ **do**
5 $Parent_1$, $Parent_2 \longleftarrow$ SelectParents($Population$, t_{size})
6 $Parents \longleftarrow$ Add($Parent_1$, $Parent_2$)
7 **end**
8 **for** $i = 0$ **to** $p_{size}/2$ **do**
9 $Parent_1$, $Parent_2 \longleftarrow$ RandomlyChooseParents($Parents$)
10 $Child_1$, $Child_2 \longleftarrow$ Crossover($Parent_1$, $Parent_2$, c_{prob}, c_{max})
11 $NewPopulation \longleftarrow$ Add($Child_1$, $Child_2$)
12 RemoveUsedParents($Parent_1$, $Parent_2$, $Parents$)
13 **end**
14 $Population \longleftarrow$ Mutate($NewPopulation$, D, m_{prob}, c_{max})
15 $DC_{best} \longleftarrow$ Evaluate($Population$, D, α)
16 **end**

4.2.4 Population

Individual encoding An individual, i.e., a DC, is encoded as a vector of single rules (genes). Each gene is a rule consisting of maximally two negated/non-negated miRNA-IDs. Here, the genotype can be changed by adding/removing negation of a miRNA, adding/removing miRNAs, changing miRNA-IDs or adding/removing

rules. A unique ID and a fitness score are assigned to each individual. Both the distributed classifier and single rules must satisfy the previously described constraints (see Section 4.2.1).

Initial population An initial population of p_{size} DCs is generated randomly, i.e., each classifier and single rule in the classifier is randomly initialized. Individuals in the population can be of a different size $c \leq c_{max}$. After randomly choosing c, every rule is generated in a few steps. First, the rule size (r_{size}) and r_{size} miRNA IDs are randomly (uniformly) chosen. Then, for each miRNA, the sign (positive/negative) is randomly assigned. The procedure is described in detail in the appendix (Algo. A1).

4.2.5 Fitness function and evaluation

Various metrics for the evaluation of binary classifiers are available (Ramola et al., 2019). However, real-world expression data is often significantly imbalanced, meaning that samples in one of the classes are overrepresented in the data. This may significantly influence the classification results (Yang et al., 2006; Rubinfeld et al., 2006; Whalen et al., 2022). The breast cancer data set described in section 3.2.6 is an example of a significantly imbalanced data set, where the negative class (representing healthy tissue environment) is heavily underrepresented. Balanced Accuracy (BACC, Eq. 4.2) is an intuitive and easily interpretable metric that allows balancing the importance of samples in both classes and evaluation of the actual performance of classifiers in case of imbalanced data (Ramola et al., 2019). Thus, as the objective (fitness) function and the primary measure of the classifier's performance, we apply balanced accuracy:

$$BACC(DC, D) = \frac{\frac{TP}{P} + \frac{TN}{N}}{2} \tag{4.2}$$

where DC is a distributed classifier, D is a given data set, P and N are the numbers of positive and negative samples in D, TP is the number of samples correctly classified as positive, and TN is the number of samples correctly classified as negative. TP and TN are threshold-dependent, i.e., they can change while applying different threshold values for a given classifier. A given DC correctly classifies a single sample if the classifier's output agrees with its annotation in the data set D (analogously to SC classifiers). To count TPs and TNs, we iterate over samples and assess the performance of a DC according to the threshold function described in section 4.2.1 and a user-specified α value. The fitness score is calculated separately for each DC in the population (Algo. 1, 2, 15). Each iteration of the GA is completed by the update of the list of the best-found solutions (DC_{best}). If the newly generated DCs perform with higher BACC than the solutions currently stored in DC_{best}, the list is cleared, and the new best DCs are added to DC_{best}. If the new DCs have identical scores as the solutions in DC_{best}, they are added to the list of the best solutions (Algo. 1, 15). In section 4.3, we discuss the influence of different thresholds on the results.

Besides balanced accuracy as the primary performance measure, to evaluate other aspects of the classifier's behaviour, we employ the following metrics: sensitivity (True Positive Rate, TPR): $TPR = TP/(TP + FN)$, specificity (True Negative Rate, TNR): $TNR = TN/(TN + FP)$, False Negative Rate (FNR): $FNR = 1 - TPR$, False Positive Rate (FPR): $FPR = 1 - TNR$ and accuracy (ACC): $ACC = (TP + TN)/(P + N)$. Sensitivity (TPR) represents the ability of the method to recognize samples belonging to the positive class correctly, and specificity (TNR) shows the

ability to recognize samples belonging to the negative class correctly. As introduced in the previous chapter, False Negative Rate (FNR) describes the probability of overlooking a positive sample, and False Positive Rate (FPR) misclassification of a negative sample as positive. Accuracy gives information about the proximity of results to true values but does not consider data imbalance.

4.2.6 Operators

Selection Parents that are potential candidates for recombination are chosen in the process of tournament selection (Algo. 1, 4-7). Tournament selection allows increasing the chance of high-fitness solutions to be selected as parents while maintaining the diversity in the population and can be efficiently implemented (Shukla et al., 2015). In each selection iteration, two parents are chosen in separate tournaments. First, t_{size} individuals are randomly chosen from the current population to participate in a tournament. The winning candidate (parent) has a higher fitness score. In each iteration of selecting two parents, the first chosen parent is not returned to the pool of possible candidates to preserve diversity. The steps are repeated to form a population of selected individuals of the pre-defined population size. See Appendix (Algo. A2) for more details.

Crossover In each crossover iteration, two individuals are randomly chosen from a population of selected parents to recombine and generate two new individuals (children). Crossover (Algo. 1, 8-13) occurs with the probability c_{prob}. A random number p is chosen to decide whether parents exchange information, and: (i) if $p \leq c_{prob}$, then the two randomly chosen parents recombine, (ii) otherwise, parents are copied to a new population without change. If chosen parents are of the same size, we perform uniform crossover (Fig. 4.3a). First, rules from the first and second parent are paired off. Then, the first rule in each pair is assigned with equal probability to either the first or second child, while the second rule is assigned to the other child. The step is repeated until all the rules from the parents are utilized, and the children consist of the same number of rules as the parents. Otherwise, if the sizes of parents differ, to preserve a chance for each rule to be exchanged, we apply an index-based uniform crossover (Fig. 4.3b). Here, the rules from the first and second parent are paired off according to a randomly chosen index specifying the position of a shorter parent in relation to the other one (see example in Fig. 4.3b). Paired rules crossover uniformly. Rules that cannot be paired (due to different sizes) may be copied to a randomly chosen child. Note that the index-based crossover may shorten the size of an individual as additional rules cannot be copied to the larger classifier. Details on the implementation of the index-based crossover may be found in the Appendix (Algo. A3 and A4).

Mutation Each classifier mutates with probability m_{prob}. Mutation (Algo. 1, 14) occurs on two levels: both rules and inputs can mutate. A rule may (i) be removed from a classifier, (ii) be added to a classifier, and (iii) be copied from one classifier to another. As mentioned before, the index-based crossover may shorten the classifier. Here, two possibilities to extend the size of a classifier are available: a new rule may be initialized and added to a classifier or copied from another classifier. These two options balance the influence of crossover on the size of classifiers. An input may (i) be removed from a rule, (ii) be added to a rule, (iii) may change the sign (i.e., become a negative or positive input respecting the constraints described in Section 4.2.1). Rules, being larger components affecting the classifier size, mutate with a

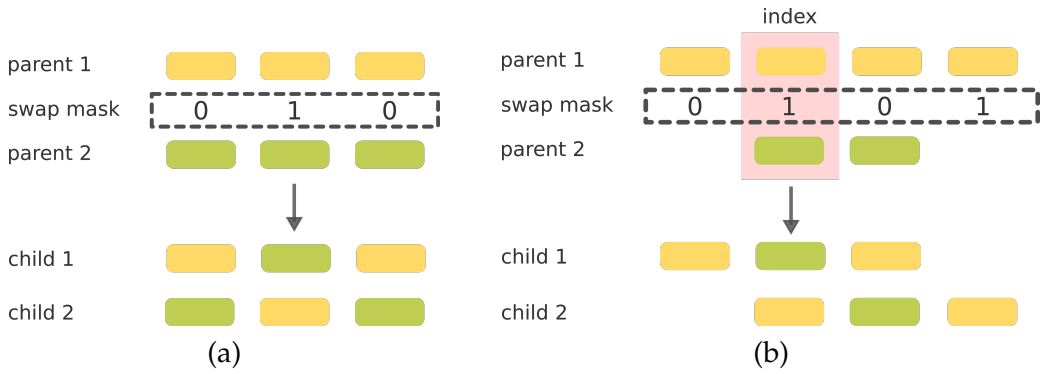


FIGURE 4.3: Crossover strategies. (a) Uniform crossover applied to individuals of the same size. (b) Index-based crossover applied to individuals of different sizes. The rules from the first and second parent are paired off according to a randomly chosen index (in red) specifying the position of a shorter parent in relation to the other one.

lower probability than inputs (0.2). Note that the maximal size of a classifier (c_{max}) must be preserved. For more details, see Appendix (Algo. 5).

4.2.7 Parameter tuning

As described in section 4.2.3, the genetic algorithm takes five different parameters describing the details of each run, which must be tuned for a particular task to obtain adequate performance. Thus, we apply a random search approach (Bergstra and Bengio, 2012) to find a suitable parameter set. A widely used alternative, grid search, relies on sequentially testing each combination appearing in a pre-defined grid of parameters. In contrast, the random search allows defining only the boundaries of the search space in which the parameter combinations are randomly sampled for testing. It has been shown that random search allows obtaining results similar to the grid search approach while significantly decreasing the computational cost (Bergstra and Bengio, 2012). We sample several combinations of parameters within a pre-defined search space. Further, we perform 3-fold cross-validation using one of the case study data sets to select the parameter set achieving the highest scores. We repeat each GA run ten times to obtain the average balanced accuracy for the validation data. Further, the parameters are employed to train distributed classifiers presented in the following section.

4.3 Case study

4.3.1 Breast cancer data

Here, we illustrate the potential of DCs in the application by performing case studies on real-world breast cancer data previously applied by both us (Becker et al., 2018) and Mohammadi et al., 2017 to the design of single-circuit classifiers (described in detail in section 3.2.6). The data sets are formatted as presented in section 3.2.6.

4.3.2 Parameters, training and validation

We have randomly chosen 300 combinations of 5 parameters in the following ranges: t_{iter}: 25 - 100, step 25; p_{size}: 50 - 300, step 50; c_{prob}: 0.1 - 1.0, step 0.1; m_{prob}: 0.1 -

1.0, step 0.1; t_{size}: 0.1 - 0.5, step 0.1 (of p_{size}). We tune the parameters for α = 0.50, corresponding to the majority vote strategy. Tuning for each value of α separately would be computationally expensive. The intermediate threshold ratio, although suboptimal, may give an estimate of well-performing parameters for other values of α. We set the upper bound on the size of DCs c_{max}=5 to preserve the maximal number of miRNA inputs as proposed for single-circuit classifiers (Mohammadi et al., 2017; Becker et al., 2018). In the process of tuning, we selected the following set of GA parameters: t_{iter} = 75, p_{size} = 200, c_{prob} = 1.0, m_{prob} = 0.3, ts = 0.1 (20 individuals).

To assess the DCs' performance, we employed 3-fold cross-validation using each of the breast cancer data sets separately. We apply the parameters selected in the tuning process to train the classifiers. We repeat the CV for eight different values of α: 0.25, 0.35, 0.40, 0.50, 0.60, 0.65, 0.75, 0.85, to evaluate the influence of the decision threshold on the classification accuracy. We run the algorithm once for each fold and set c_{max}=5. At the end of the run, scores for the single best-performing and shortest classifier are recorded.

4.3.3 Performance evaluation

The results for best performing α values are presented in Table 4.1 (the complete results for particular thresholds may be found in the Appendix, Table S4.1). If more than one threshold resulted in identical BACC values, we present results for a threshold with the highest BACC recorded in training. In the case of equal training BACC values, we present exemplary results for a chosen threshold. Table 4.1 includes the values of α and performance scores. Solutions recorded for all folds corresponding to the results may be found in Table 4.2.

Dataset	α	$BACC_{tr}$	TPR	TNR	ACC	$BACC_{val}$
All	0.50	0.98	0.92	0.92	0.92	0.92
Triple-	0.85	0.98	0.92	0.75	0.89	0.83
Her2+	0.75	0.96	0.99	0.61	0.94	0.80
ER+ Her-	0.50	0.93	0.90	0.64	0.82	0.77
Cell Line	0.25	1.00	1.00	1.00	1.00	1.00

TABLE 4.1: Results of 3-fold cross-validation. For the breast cancer All data set we found DCs performing with identical BACC for two α values (0.50, 0.60) and for ER+ Her- for six different α values (0.35, 0.50, 0.60, 0.65, 0.75, 0.85). Metrics: $BACC_{tr}$ - average balanced accuracy recorded for training data in the 3-fold CV, TPR - average True Positive Rate recorded for validation data in the 3-fold CV, TNR - True Negative Rate, ACC - accuracy, $BACC_{val}$ - balanced accuracy recorded for the validation data.

TABLE 4.2: Solutions for all folds of breast cancer data sets corresponding to the results presented in Table 4.1. The rules repeating between folds are in bold.

Dataset	Solution
All fold 1	NOT hsa-miR-19b-2, **NOT hsa-miR-145**, NOT hsa-miR-193a-5p, **hsa-miR-200c**
fold 2	**NOT hsa-miR-378**, NOT hsa-miR-126, **hsa-miR-200c**, hsa-miR-142-3p
fold 3	**NOT hsa-miR-378**, **hsa-miR-200c**, **NOT hsa-miR-145**, NOT hsa-miR-451-DICER1
Triple- fold 1	**NOT hsa-miR-320-RNASEN**
fold 2	**NOT hsa-miR-451-DICER1**, NOT hsa-miR-145, NOT hsa-miR-423-3p, hsa-miR-24-1
fold 3	**NOT hsa-miR-320-RNASEN**, **NOT hsa-miR-451-DICER1**, hsa-miR-99a, (hsa-miR-24-1) AND (NOT hsa-miR-378)
Her2+ fold 1	NOT hsa-miR-451-DICER1, (NOT hsa-miR-320-RNASEN) AND (NOT hsa-miR-99a)
fold 2	(NOT hsa-miR-451-DICER1) AND (NOT hsa-miR-125b-1), **hsa-miR-21**
fold 3	**hsa-miR-21**
ER+ Her- fold 1	hsa-miR-378, (hsa-miR-21) AND (NOT hsa-miR-451-DICER1), NOT hsa-miR-320-RNASEN
fold 2	NOT hsa-miR-125b-1
fold 3	(hsa-miR-21) AND (NOT hsa-miR-320-RNASEN)
Cell Line fold 1	hsa-miR-425, NOT hsa-miR-143
fold 2	NOT hsa-miR-199a-2-5p, NOT hsa-miR-221
fold 3	NOT hsa-miR-146a, NOT hsa-miR-143

[h]

High BACC values obtained for the training data sets and the final average population BACC values (0.91) show that the algorithm converges on good solutions, resulting in high-performing DCs. The BACC values measured for the validation data are particularly higher for the largest (*All*) and the smallest (*Cell Line*) data sets than the intermediate-size ones. As the *All* data set consists of the largest number of learning examples, it potentially contains more information about the miRNA expression patterns. The *Cell Line* data set includes six different miRNAs that perfectly separate samples (Becker et al., 2018). Therefore, excellent performance was expected for this particular data set. For the data sets *Her2+* and *ER+ Her-*, the number of relevant miRNAs is lower than for the other data sets (see Table 3.1). Therefore, the search space of feasible solutions is also substantially decreased compared to the other data sets. The accuracy is higher than BACC for all data sets. However, the metric is not sensitive to data imbalance and seems to overestimate the classifiers' performance.

The sensitivity (TPR) is high for all data sets meaning that the method successfully classifies samples belonging to a positive class. In all the data sets except for the *Cell Line*, the positive class is over-represented in terms of sample abundance. Hence, the algorithm has access to a large number of learning examples that possibly better describe the patterns behind the positive class. The specificity (TNR) is particularly decreased for the Her2+ and ER+ Her- subtypes. Note that the data sets are substantially imbalanced, i.e., the negative class is strongly underrepresented. Hence, the number of learning examples is significantly smaller. Also, even a few false-positive errors result in substantially lower specificity. In terms of cell classifiers, this may increase the toxicity of the therapy as the drug would be released in misclassified healthy cells.

The best performing α values differ among the data sets. For the largest one, α is equal or not much higher than 0.50. According to the average number of rules in DCs for the *All* data set (see Table 4.2), the threshold 0.50 corresponds to exactly 50% of rules (two out of four). The data sets of intermediate sizes (*Triple-* and *Her2+*) favoured two more extreme α values. Here, the thresholds require all the rules to classify the sample as positive to trigger the response in the case of all classifiers for both data sets. For the ER+ Her- subtype, several α values returned identical results (Appendix, Table S4.1). For the smallest data set, the lowest α value resulted in the highest BACC. However, taking the size of DCs into account, the threshold is, in fact, equal to 0.5 in terms of the classifier's decision. The α parameter seems to be data-related and should be tuned for a given data set to increase the performance of DCs.

The best performing α values presented in Table 4.1 are collected based on BACC recorded for the validation data. As validation data should represent a fraction of unseen samples simulating the real-world environment, the α cannot be selected in such a manner in practice. In Table 4.3, we present the average performance of classifiers for (i) all employed thresholds and (ii) thresholds achieving the highest BACC on training data. This allows estimation of the threshold performance in terms of generalization to novel samples. The scores for both above-mentioned cases do not differ substantially. The largest variance in performance is recorded for the smallest *Cell Line* data set. Although the classifier's threshold may be estimated using training data, the performance is substantially lower for most of the data sets in comparison with the highest achieved scores presented in Table 4.1. We comment on this further in the discussion.

Applying a certain threshold caused a slight shift in the rates of certain types of errors. Here, we analyze false positive rates (FPR) and false negative rates (FNR)

Dataset	Thresholds	$BACC_{tr}$	BACC_{val}	σ_{te}
All	all	0.98	0.86	0.04
	best	1.00	0.84	0.00
Triple-	all	1.00	0.77	0.03
	best	1.00	0.76	0.02
Her2+	all	0.96	0.76	0.02
	best	0.96	0.76	0.02
ER+ Her-	all	0.93	0.76	0.02
	best	0.93	0.76	0.02
Cell Line	all	1.00	0.87	0.07
	best	1.00	0.87	0.07

TABLE 4.3: Average performance of classifiers recorded for (i) all employed thresholds and (ii) thresholds achieving the highest BACC on training data (All: 0.25, Triple-: 0.25, 0.40, 0.50, 0.60, 0.65., Her2+: all except for 0.25, ER+, Her-: all except for 0.25, Cell Line: all)

observed among all data sets for two extreme α values. In the case of a low threshold (0.25, FPR = 0.34), the shift is displayed towards misclassification of the negative samples compared to the highest employed threshold (0.85, FPR = 0.27). The high threshold (0.85, FNR = 0.13) causes more frequent misclassification of positive samples in comparison to the lowest one (0.25, FNR = 0.04). This could be related to the multi-circuit classifier's structure. The higher the decision threshold α is, the more rules must agree with the prediction *'cancerous'*. In extreme cases (very high α), all the rules must identify a sample as *'cancerous'*, which may result in increased FNR. Complete information about FPR and FNR for different thresholds may be found in Table 4.4. As employed breast cancer data sets are significantly imbalanced, the influence of certain thresholds on the shift should be further investigated using alternative data sets.

TABLE 4.4: Average FPR and FNR values for different thresholds (for all datasets).

Threshold	FPR	FNR
0.85	0.27	0.13
0.75	0.23	0.11
0.65	0.31	0.11
0.60	0.26	0.12
0.50	0.26	0.12
0.40	0.35	0.11
0.35	0.34	0.04
0.25	0.34	0.04

The tests were performed using Allegro CPU Cluster provided by Freie Universität Berlin. The average run-time is 45 min for one cross-validation fold of the largest data set employed in the case studies. Thus, the tests may be performed on a personal computer. However, the breast cancer data sets consist of up to 180 samples and up to 60 relevant miRNAs. Therefore, extended scalability tests would help

estimate the run-time limits of the method and optimize the computational cost of the implemented code.

4.3.4 Analysis of input viability

In this section, we analyze miRNA inputs that occur in two exemplary classifiers. We chose the best-performing classifiers for the largest data set (*All*), representing all subtypes, and the smallest *Cell Line* data set.

For breast cancer *All*, two different α values resulted in the highest BACC. The classifiers for each cross-validation fold in the data set are identical for both α and are of the same size $c = 4$. In this case, the applied α does not change the threshold function between both values (0.50 and 0.60), i.e., for all data sets, at least two rules must output 1 to classify a cell as positive. Here, we present a DC found for the third cross-validation fold of the All data set (i.e., the fold on which the algorithm performed best, with $\text{BACC}_{tr} = 1.00$, BACC = 0.95). The classifier consists of 4 different 1-input rules: (1) NOT miR-378, (2) miR-200c, (3) NOT miR-145, (4) NOT miR-451-DICER1. We analyzed the miRNAs and found that all of them may be relevant for cancer sample classification. miR-378, miR-145, miR-451-DICER1 are described as down-regulated in breast cancer (Ding et al., 2017; Farazi et al., 2011), e.g., the study by Ding et al., 2017 has shown that underexpression of miR-145 is related to increased proliferation of breast cancer cells. Also, miR-378 occurred as down-regulated in the best 1-input single-circuit classifier presented in Chapter 3 (Becker et al., 2018) for the same data set. miR-200c is marked as up-regulated in breast cancer in Sánchez-Cid et al., 2017.

Another classifier we present is a DC for the third cross-validation fold for the Cell Line data set. The classifier consists of two rules: (1) NOT miR-146a, (2) NOT miR-143. Both miRNAs (negated) were previously presented as parts of classifiers described in Chapter 3 (Becker et al., 2018) as perfectly separating the positive and negative samples. For most of the α values, the performance of found *DC*s is significantly lower for this particular fold in the *Cell Line* data set (BACC = 0.50). A perfect classifier of size two performing with BACC = 1.00 on both training and testing data was found with $\alpha = 0.25$ (equivalent to 0.50), i.e., one of 2 rules must output 1 to classify the cell as positive. We found that both miR-146a and miR-143, are described as down-regulated in breast cancer (Li et al., 2015; Ng et al., 2014).

4.3.5 Comparison to other methods

We optimized single-circuit classifiers with the ASP-based approach described in Chapter 3 (Becker et al., 2018) by performing 3-fold cross-validation employing the breast cancer data sets and using identical folds as in the case of training distributed classifiers. The ASP algorithm's objective function is based on minimizing the total number of classification errors. Note that the ASP method may return several optimal classifiers if such exist. Different combinations of FPs and FNs influence balanced accuracy. Thus, we trained the classifiers using balanced accuracy to increase the chance of ASP performing well in comparison. Here, we do not compare our results to Mohammadi et al., 2017, as their approach did not perform better than the ASP-based approach described in Chapter 3 (Becker et al., 2018) in terms of binary classification. For optimization of single-circuit classifiers, we employ the constraints described in 2.4.

The DC-based method outperformed the single-circuit approach in 3 of 5 case studies. For two other data sets, the resulting BACC (validation) values are either

TABLE 4.5: Comparison of results of 3-fold cross-validation for the ASP-based approach proposed by Becker et al. Becker et al., 2018 and for the GA (as in Table 4.1).

Dataset	Method	Sensitivity	Specificity	ACC	BACC	$BACC_{train}$
All	GA	0.92	0.92	0.92	**0.92**	0.98
	ASP	0.96	0.47	0.93	0.72	0.92
Triple-	GA	0.92	0.75	0.89	**0.83**	0.98
	ASP	0.89	0.44	0.83	0.67	0.96
Her2+	GA	0.99	0.61	0.94	0.80	0.96
	ASP	1.00	0.61	0.95	**0.81**	0.96
ER+ Her-	GA	0.90	0.64	0.82	**0.77**	0.93
	ASP	0.90	0.64	0.82	**0.77**	0.93
Cell Line	GA	1.00	1.00	1.00	**1.00**	1.00
	ASP	0.83	1.00	0.93	0.92	1.00

identical (*ER+Her-*) or very similar (*Her2+*). This may imply that further improving classifier performance for those data sets is not possible with the currently applied techniques. The training BACC values are also significantly higher for the DC-based approach. Note that the DC-based design method explores a different search space than the single-circuit approach. Although single circuits are also allowed as 1-rule classifiers, their complexity is substantially lower in comparison to SC classifiers. Additionally, ASP returns globally optimal solutions, i.e., it adjusts the classifier perfectly to the training data, which may cause overfitting. Although the classifiers obtain high BACC on the training data (average for all data sets: 0.95), the classifiers may be too specific to perform well on the validation data. However, as mentioned before, the best-performing values of α for the distributed classifiers were chosen based on the BACC recorded on validation data. Considering average results presented in Table 4.2 (including also worst performing thresholds) for data sets All and Triple- the performance is substantially higher. For the remaining data sets, the results are similar or slightly worse.

In Chapter 3, we presented a scalability study demonstrating the ASP's performance regarding run-time. As mentioned before, the ASP-based approach optimizes classifiers of a different architecture than the GA-based method. Also, at the moment of comparison, ASP-based optimization requires a substantial amount of manual input. Thus, the run times of both approaches are not easily comparable. We revisit this issue in the next chapter.

4.4 Discussion

In this chapter, we introduced a proof-of-concept approach for cell classifier design by re-formalizing the concept of distributed classifiers previously proposed by Didovyk et al., 2015 in the context of miRNA-based cell classification. The presented case study demonstrates the DC's ability to perform classification on real-world cancer data. The proposed algorithm allows optimizing classifiers that achieve high accuracy in training. Further, the cross-validation study indicates that the optimized DCs classify unknown data with moderate to high accuracy. However, compared with single-circuit classifiers, multi-circuit devices show the potential to increase cell classification accuracy, in particular, boosting the classifiers' specificity.

The problem of designing robust classifiers begins with the initial data processing. The breast cancer data sets employed in the case study are significantly imbalanced, which can impede the classifiers' training (Yang et al., 2006; Whalen et al., 2022). Although we apply an objective function that partially allows overcoming this issue, one may consider applying data balancing methods such as under- and oversampling or weighted schemes that balance the sample importance (Haixiang et al., 2017). Alternatively, the classifiers could be trained on balanced data sets to eliminate the influence of class over-representation on the results (Whalen et al., 2022). We address this issue in the following chapter by employing balanced data and re-evaluating the classifiers in different scenarios. Further, as in the case of the ASP-based approach, the proposed method for DC design requires discretized data as an input, limiting its re-usability. Thus, in the next chapter, we introduce a preprocessing step allowing for binarizing the data would be advantageous to further applications, e.g., designing classifiers based on gene expression data. Note that the motivation behind this study was to showcase the potential of multi-circuit designs and present a proof-of-concept algorithm for their optimization. In the next chapter, we address this limitation and introduce a binarization strategy as a part of the workflow.

We compared the performance of multi- and single-circuit classifiers and obtained higher (or similar) performance. Although the DCs perform better on the largest and the smallest data sets than on the intermediate-size ones, the results obtained for both methods for Her2+ and ER+Her- data sets are almost identical. This suggests that significant improvement is not possible for those data sets with currently applied approaches. We also compared the average performance of DCs obtained for different thresholds with SC classifiers. The increased or comparable performance holds for most of the data sets. The improvements in binary classification likely result from applying a different strategy to cell classifier design. Here, a single-circuit decision is complemented by a collective classification of multiple circuits. Thus, the DCs may be more resistant to data noise than single-circuit classifiers.

The parameter set applied in training was tuned using the breast cancer *All* data set that comprises sub-type samples (Triple-, ER+ Her- and Her2+), causing information leakage between the training and validation fractions of the data. This could have resulted in overestimating the generalization error, indicating the classifiers' performance on unseen data (Whalen et al., 2022). Further, choosing a suitable decision threshold seems to pose a particular challenge. Using training scores as a threshold indicator results in overfitting visible in the cross-validation. Alternatively, an additional validation data set should be considered to select the best-performing parameters. However, in this case, separating a fraction of data to tune the parameters and creating a hold-out test data set was challenging due to a low number of negative samples. We address this issue in the next chapter by refining our training and evaluation strategies and minimizing information leakage. Also, we incorporate the decision threshold as a part of the classifier instead of using it as a tunable parameter. The ASP approach does not require parameter tuning, which gives a significant advantage in this case. However, the results obtained for the ASP method are substantially lower, dropping below 0.68 for the Triple- data set.

Although DCs are not yet applied in cancer cell classification, the approach should be further investigated. DCs are designed based on available building blocks that are, in fact, single-circuit classifiers. Mohammadi et al., 2017 presented a biochemical model of a single-circuit classifier that allows manipulating the output compound concentration. Thus, the biological output threshold for a given classifier can

be adjusted to perform the classification in living cells. As the on-off single-circuit response may be regulated on the biological level, the sum of their outputs should also be adaptable for a given DC, e.g., by introducing a signal amplification step after reaching the threshold of an intermediate compound.

Chapter 5

Finding Robust Classifiers

Contribution note

A part of the work presented in this chapter has been published in https://www.biorxiv.org/content/10.1101/2020.05.13.092908v1 (Nowicka and Siebert, 2020). The second version of the manuscript is under preparation and will soon be available on biorxiv. The employed data, Jupyter Notebooks allowing reproducibility of data processing and code are stored in a GitHub repository at: https://github.com/MelaniaNowicka/RAccoon.

The UMAP visualisations are generated using a script written by Melania Nowicka and Jakub Bartoszewicz as a part of a study: JM Bartoszewicz, F Nasri, M Nowicka, BY Renard, Detecting DNA of novel fungal pathogens using ResNets and a curated fungi-hosts data collectionBartoszewicz et al., 2022.

5.1 Introduction

In the previous chapter, we have shown the potential of distributed classifiers to recognize breast cancer samples in two scenarios. In the first case, the positive group consisted of mixed subtype cancer samples, whereas in the second, the positive class was represented by a single cancer subtype or cell line. According to the presented studies, distributed classifiers substantially improved sample classification compared to single-circuit classifiers. However, the employed data and training-and-validation strategy provide only a narrow insight into the performance of the multi-circuit classifiers facing different environments.

The breast cancer case study is limited to a single cancer type, providing no information about the circuit's performance in other cancers. The data is also significantly imbalanced (imbalance ratio reaching 15 indicating that the positive class comprises 15 times more samples than the negative class), hindering the training of well-generalizing classifiers (Whalen et al., 2022). The under-represented negative class is limited to 11 samples, meaning that after division into cross-validation folds, the training fraction comprises an even smaller number of control samples. Here, the scarcity of negative samples in training may result in the misclassification of healthy cells, increasing the toxicity of the therapy.

Another limitation is the lack of data for which the ground truth about the features is known, namely, the truly up- and down-regulated markers. This was partially shown in the breast cancer study in which our classifiers recovered features also described by Farazi et al., 2011 as differentially regulated. Identifying relevant features can significantly advance the evaluation of classifiers by indicating whether

the trained circuits pinpoint truly valuable feature candidates. However, a more extensive study performed in a controllable environment could bring more insights into the true signal recovery.

Also, the accessibility and applicability of the algorithm proposed in the previous chapter are limited, as the approach operates exclusively on binarized data and is not flexible in terms of the data input. The binarization approaches allow the profiling of the feature expression levels in an interpretable manner. However, the advantage of binarization in categorizing the continuous data makes it also prone to information loss. Depending on the discretization approach, one can partially retrieve missing information, which can be further employed to optimize classifiers by pointing toward more robust feature candidates. For instance, using the information about the discriminative power of the features explicitly in the objective function can facilitate finding good candidates for the classifier inputs.

Another relevant issue often neglected in the study design is the information leakage between the training and test fractions of the data. Recently, Whalen et al., 2022 described various machine learning pitfalls common for genomics studies, which are, in fact, inherent in most classification problems where machine learning is applied to biological (and non-biological) data. The proper study design is crucial for estimating the generalization error and allows for predicting the classifier's performance while facing unseen data. The examples included in the training data describe only a part of the environment the classifiers must face in practice. Thus, the appropriate estimation of classifier behaviour tackling novel data is essential for real-world application. In the breast cancer study, the data sets were previously processed by Mohammadi et al., 2017. The processing comprises normalization and binarization of miRNA expression and is followed by data split into cross-validation folds. However, it is recommended to perform the data division at first, if possible, on raw data and apply the necessary processing afterwards (Whalen et al., 2022). Otherwise, information can be passed between the training and test fractions, making the test samples simpler to recognize. This may result in a more optimistic generalization error than the real-world performance and decrease the overall robustness of the circuits in terms of classifying novel data (Whalen et al., 2022).

In this chapter, we address the issues mentioned above. We extend our workflow with a data processing step, particularly the binarization procedure that may be applied to different data types. We employ an approach that allows threshold-based feature selection and provides easily interpretable scores indicating the feature robustness. We further employ this measure to design and optimize cell classifiers. We extend the objective function to take not only the accuracy of the entire classifier but also the robustness scores of particular miRNAs into account. Here, we score the features based on their discriminative power and employ the assigned scores directly in the optimization process. In terms of the study design, we focus on the train-test strategy, in particular on modelling different scenarios simulating various data perturbations. First, we generate synthetic data sets to create a controllable environment for performance evaluation. We establish a simulated data study that allows testing classifiers to face pre-defined experimental conditions and evaluating the classifier's features with regard to the ground truth. Here, we set up an evaluation strategy to assess classifiers' performance against the noise of different intensities and types. Further, we simulate data of a different size to capture the algorithm's run-time. We also gather four miRNA expression profiles representing liver and lung cancer. We search for balanced data sets to build reliable training example sets. We train classifiers in two scenarios by performing joint and separate normalization of training and test data and compare multi- and single-circuit designs. Finally, we perform a

cross-platform evaluation of trained classifiers and analyze exemplary designs.

5.2 Data

5.2.1 Discretization

We adopted the discretization method proposed by Wang et al., 2014 to binarize the miRNA expression profiles. The authors demonstrated that their approach outperforms other commonly used supervised discretization methods when applied to the classification of cancer gene expression data of different types, e.g., cancer vs control or subtype A vs subtype B. (Wang et al., 2014; Gallo et al., 2016b). The approach defines the expression pattern of each feature in the data across all samples by employing information about the sample labels. Each feature is discretized into one, two or three states by finding zero, one or two cut-points, respectively, depending on its expression in each class. According to such an approach, a feature may be (i) in the same regulatory state in both classes (Figure 5.1A), (ii) up-regulated in one class and down-regulated in the other class (Figure 5.1B) or (iii) be both up- and down-regulated in one class and non-regulated in the other class (Figure 5.1C). In the latter case, the feature is discretized into three states: -1, 0 and 1, which is not compatible with the Boolean representation of a classifier and, what follows, with the classifier's functional structure. Thus, in terms of the cell classifier application, only miRNAs that follow the second pattern (B) are valuable candidates for the inputs. We refer to such miRNAs as relevant and the remaining ones (A and C) as non-relevant features.

To find the thresholds, the expression range of each miRNA (increasing from the minimal to the maximal expression value for a given sample) is divided into m ($m \geq 50$) left-side-half-open intervals of equal size, where $v_i = (-\infty, l_i]$ is the i-th half-open interval and $l_i = 1, 2, ..., m$ are the upper-boundaries of the m intervals. Here, $v_m = (-\infty, l_m]$ comprises all expression values. For each of the intervals the class distribution diversity CDD($\mathrm{v_i}$) is calculated according to Eq.5.1:

$$CDD(v_i) = \frac{n_1(v_i)}{N_1} - \frac{n_2(v_i)}{N_2} \tag{5.1}$$

where $n_1(v_i)$ and $n_2(v_i)$ represent the numbers of samples belonging to the first or the second class in the interval v_i. N_1 and N_2 are the total numbers of samples in each class. This gives information about whether in the interval v_i one of the classes is overrepresented, and if yes, which one it is, according to following cases: if (i) $CDD > 0$, class 1 is overrepresented, (ii) $CDD = 0$, both classes are equally represented, (iii) $CDD < 0$, class 2 is overrepresented. Then, among all the intervals, two with the maximum and minimum CDDs ($\mathrm{CDD_{max}}$ and $\mathrm{CDD_{min}}$) are found and the global class distribution diversity of a feature $\Delta_{feature}$ can be calculated according to Eq.5.2:

$$\Delta_{feature} = |CDD_{max} - CDD_{min}|. \tag{5.2}$$

Here, $\Delta_{feature} \in [0, 1]$ describes the overall capability of a feature to distinguish between two classes. The discriminative power increases with $\Delta_{feature}$, i.e., if the classes are perfectly separated $\Delta_{feature} = 1$. We employ this measure to assess the robustness of miRNAs as classifier inputs (described further in section 5.3.2). The authors use $\Delta_{feature}$, CDD_{max}, CDD_{min} and two other tunable parameters α and λ to find the discretization cut-points based on three criteria described in detail in Wang et al., 2014.

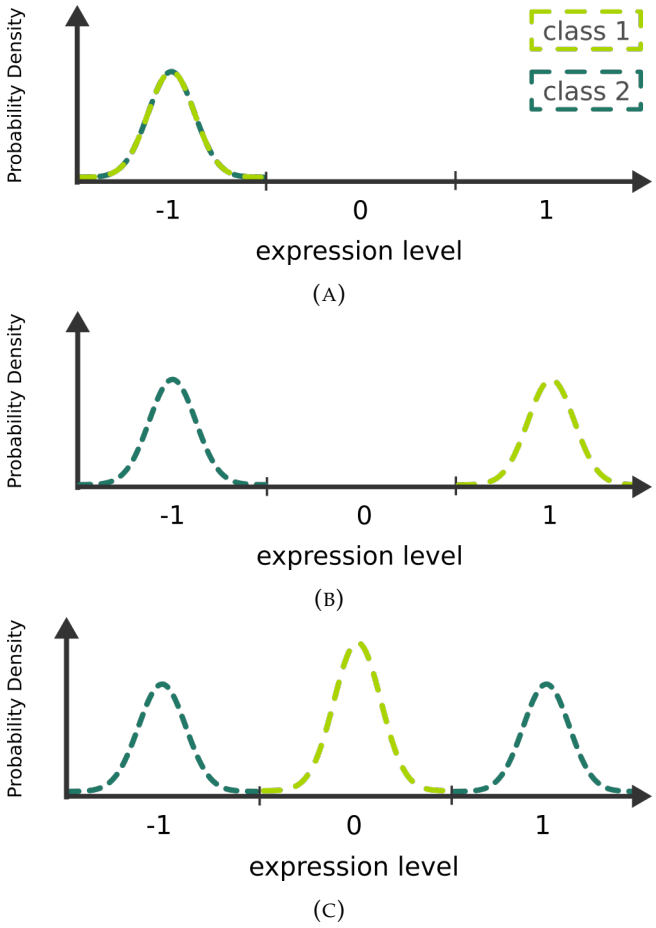


FIGURE 5.1: Three expression patterns proposed by Wang et al., 2014. Down-regulation is marked as -1, no regulation as 0 and up-regulation as 1, respectively. A gene may be (A) in the same regulatory state in both classes, (B) up-regulated in one class and down-regulated in the other class or (C) be both up- and down-regulated in one class and non-regulated in the other class. We discard all miRNAs following (A) or (C) patterns from the data set. Figure adapted from Wang et al., 2014.

In this study, we refer to the discretization parameters as m_{bin}, α_{bin} and λ_{bin}. α_{bin} is a lower-bound on the $\Delta_{feature}$ that allow distinguishing between non-regulated (Figure 5.1A) and differentially regulated features (Figure 5.1B and C). Thus, α_{bin} can be used as a cut-off for relevant and non-relevant features. The λ_{bin} parameter decides whether the miRNA is discretized into two or three states (Figure 5.1B and C, for details see Wang et al., 2014). Based on results employing four real-world cancer data sets and different values of α_{bin} and λ_{bin} the authors demonstrated that $\alpha_{bin} = 0.5$ and $\lambda_{bin} = 0.1$ may be applied as default parameter values in practice. We transform the continuous data described in the further sections into discrete profiles employing the above-described strategy. This allows converting the data set of continuous values into a binarized data set $D = (S, A)$ defined in section 2.3. We also remove non-relevant features from further analysis to reduce the data sets' size and increase the computation's efficiency.

5.2.2 Simulated expression data

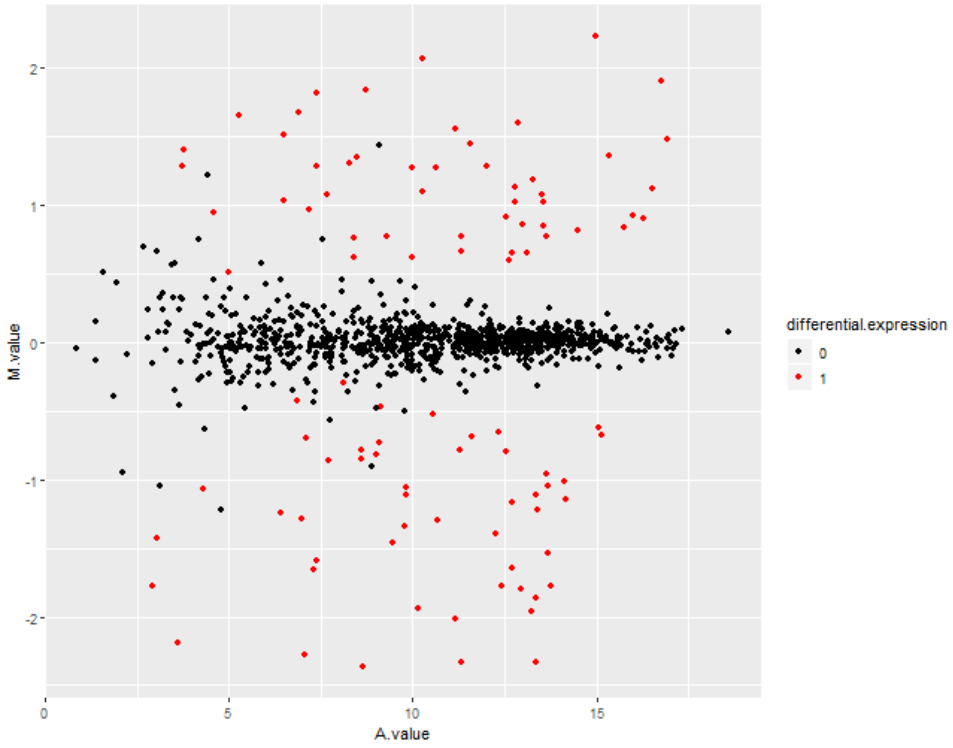
Data simulation

To simulate synthetic data sets, we employ the compcodeR package developed by Soneson, 2014. CompcodeR is a benchmarking Bioconductor library designed to simulate differential gene expression RNA-seq experiments using various parameters. This allows controlling the setting of each experiment regarding the number of samples in each class and the fraction of truly differentially regulated genes, as well as the noise level in the data (Soneson and Delorenzi, 2013). As a default, compcodeR uses the Negative Binomial distribution to simulate RNA-seq counts. The distribution parameters, i.e., the mean and the dispersion, are estimated based on real-world gene expression data sets (Soneson and Delorenzi, 2013). Depending on the purpose of the data set, we employed different simulation parameters. To estimate the performance of classifiers regarding accuracy and robustness to noise, we generated data consisting of (i) 200 samples in total, 100 samples per class, and (ii) 1000 genes, of which 10% are differentially expressed, where (iii) 50% of the differentially expressed genes are up-regulated. Here, the number of samples and features approximately corresponds to the maximal size of cancer data sets employed in cancer case studies (described in section 5.2.3). The fraction of differentially expressed and up-regulated genes comes from the setting proposed by Soneson and Delorenzi, 2013.

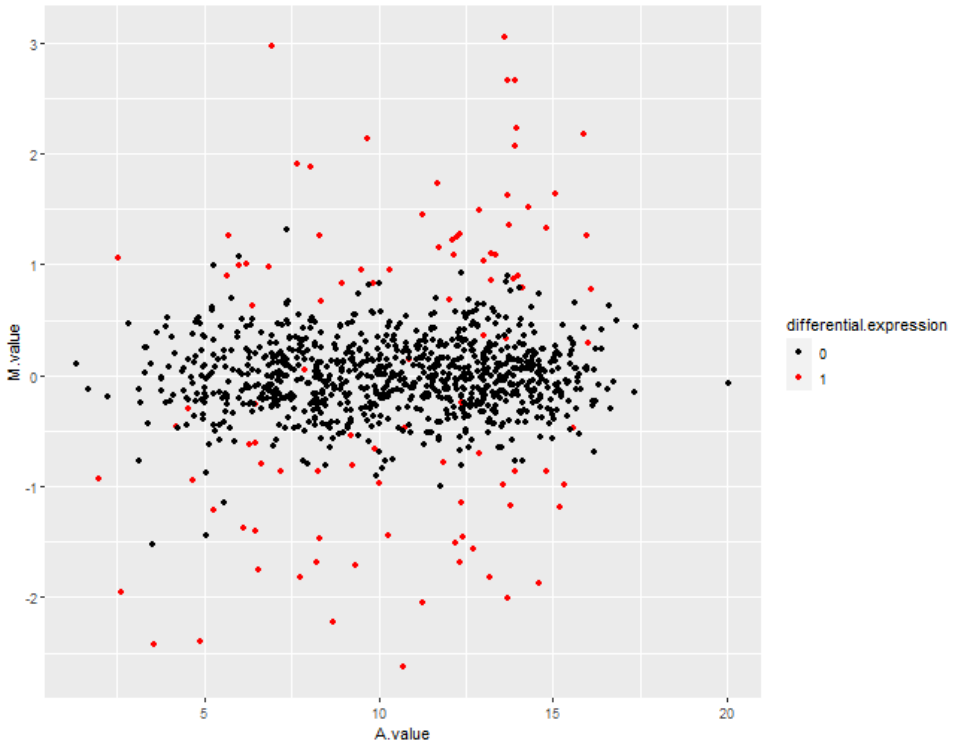
Further, we introduce different types of variations in the data. First, we test the classifiers against the occurrence of random outlier values among the read counts. Here, each observed count is multiplied or divided by a random number between 5 and 10, with an equal probability of increased or reduced value. We introduce the outliers with probabilities varying between 0.0 and 0.5 with step 0.1 labelling data sets as SDR0, SDR10, SDR20, SDR30, SDR40 and SDR50 (Simulated Data Random, SDR), where the number indicates the probability of random outliers allowed in the simulation. All the other parameters are set to default. The MA plots visualising the differences between all the data sets are presented in Figures S5.1. The plots represent log2 fold changes (M, vertical axis) versus the average expression signal (A, horizontal axis) for all samples in each data set. Red dots correspond to truly differentially regulated features, and black ones to non-regulated features. Increasing noise is visible in the sparsity of non-regulated features (black dots) along the horizontal axis (A values). In particular, in Figure 5.2B, the non-regulated features blend with several truly differentially regulated ones compared to Figure 5.2A.

We also test two other scenarios by altering the distribution parameters. First, we generate a data set with different dispersions between the two conditions using the Negative Binomial distribution (SDDISP). This allows evaluating the DC's performance against the different spread of expression values between the classes. Then, we generate a data set where 50% of the genes are simulated following the Negative Binomial distribution and 50% are simulated following the Poisson distribution (SDP50) to assess the DC's robustness against features following different distributions across the samples. The MA plots of SDDISP and SDP50 data sets are shown in Figure S5.2.

Finally, we generate a more extensive data set (Simulated Data Large, SDL) as a baseline to reduce the number of features and samples afterwards and capture the run-time of the employed optimization methods in terms of different data sizes. For this purpose, we simulate a data set consisting of 200 samples (balanced) and 10000 features, of which 80% are differentially expressed (50% of the differentially expressed genes are up-regulated), with a random outlier probability of 0.5. Here,



(A) SDR0



(B) SDR50

FIGURE 5.2: MA plots for the outlier simulated data sets: SDR0 and SDR50. Red dots represent truly differentially expressed features, and black dots - are features that are not differentially regulated. Plots generated with the compCode R package (Soneson, 2014).

we simulate a data set abundant in differentially regulated features to extend the data size. We employ the SDL data set to create subsets of data with various number of features and samples. Further steps of data set reduction are described in the following sections.

To test the performance of distributed classifiers, we employ 5-fold cross-validation for each data set (except for SDL, of which the primary purpose is to measure the algorithm's run-time). The SDL data set is split into single training (80%) and test (20%) fractions. The data division is performed directly after the simulation.

Normalization

The synthetic data sets are normalized with the TMM (trimmed mean of M-values) normalization method available in the edgeR package (Robinson et al., 2009). TMM normalization was shown to maintain a low false-positive rate in terms of detecting differentially regulated genes while facing count variations in the data (Dillies et al., 2013). The normalization strategy assumes that most of the genes in the data are not differentially expressed (Dillies et al., 2013). The fraction of differentially regulated features in the simulated data sets does not exceed 10% (except for the SDL data set), making the TMM normalization applicable. Usually, the normalization factors are calculated using a single reference sample chosen from all samples in the data set (Robinson et al., 2009). In this study, we normalize the training and validation data sets separately to avoid information leakage (Whalen et al., 2022). Hence, the reference sample was chosen based on the training data set and then used to normalize the training and testing data sets. The procedure, not directly available within the edgeR package, is implemented in R and accessible within the code repository.

Discretization

The data sets are binarized using the discretization method by Wang et al., 2014 described in section 5.2.1. The training and testing data sets are discretized separately, as in the case of data normalization, to avoid information leakage. First, the discretization thresholds of each feature in the data set are calculated based on the training samples with the default parameters estimated by Wang et al., 2014: m_{bin}=50, α_{bin}=0.5, λ_{bin}=0.1. Then, the thresholds are applied to discretize the testing data set. All non-relevant features are filtered out from the training data. For each feature in the training data, the global class distribution diversity $\Delta_{feature}$ is calculated and stored.

Table 5.1 contains the average (across all folds) numbers of features selected in the process of discretization, the average values of $\Delta_{feature}$ over each of them, as well as the average values of $\Delta_{feature}$ for all features in the data set before selection. According to the criteria proposed by Wang et al., 2014, the cut-off for relevant features lies at $\Delta_{feature} \geq 0.5$. This allows separating features with low values of $\Delta_{feature}$ that are abundant across all data sets as shown in Table 5.1. The number of relevant features following the two states' regulation pattern drops with the growing number of outliers in the data set, which is also true for the average $\overline{\Delta}_{feature}$. This is expected, as perturbed data sets contain less distinctly regulated features. It is particularly evident in the case of the SDR40-50 data sets, where the probability of the outlier occurrence is the most elevated. The SDP50 data set seems to contain many features that follow clear-cut patterns (also shown in Figure S5.2B). This feature selection approach allows selecting of potentially valuable features, reducing the search space

(by removing non-relevant features) and thus also decreasing the run-time of the computation. Note that extending the binarization threshold may substantially reduce the number of features in the data or eliminate them completely.

We compared the features selected in the discretization process with the ground truth-regulated features. For most data sets (except for the SDDISP data set), the discretization allowed the selection of only truly differentially expressed features as relevant. The different levels of dispersion between the two conditions make it slightly more difficult for the discretization approach to recover the ground truth. However, the majority of the selected features (87%) belong to the truly regulated fraction.

Dataset	Features	$\overline{\Delta}_{feature}$ (all)	$\overline{\Delta}_{feature}$
SDR0	65	0.207	0.762
SDR10	57	0.171	0.705
SDR20	49	0.166	0.658
SDR30	35	0.162	0.611
SDR40	19	0.157	0.574
SDR50	6	0.155	0.531
SDDISP	92	0.361	0.718
SDP50	84	0.229	0.899

TABLE 5.1: The number of relevant features, average values of $\Delta_{feature}$ for all features $\overline{\Delta}_{feature}$ (all) and over differentially regulated features after binarization ($\overline{\Delta}_{feature}$).

Data set reduction

To record the run times, we prepare subsets of the SDL data set by reducing the number of features and samples. This allows isolating the influence of the data size, as generating independent data sets results in separate problem instances. First, we randomly reduce the number of samples (Simulated Data Samples, SDS) in the normalized training fraction of SDL (Simulated Data Large) data set to 60% (120 samples, SDS120), 40% (80 samples, SDS80), 20% (40 samples, SDS40) and 12% (24 samples, SDS24) as well as keep the original size training data set (160 samples, SDS160). The test fraction remains unaltered (40 samples per data set). With SDS data sets, we aim at assessing the algorithm's performance in terms of the reduced number of training examples. Further, we employ the original SDL data set to generate subsets with different numbers of features (Simulated Data Features, SDF). The number of features can be reduced in the discretization process by decreasing the lower bound on the α_{bin} defining the threshold between relevant and non-relevant features. This results in a different number of features in the data set, enabling assessment of the algorithm's scalability against different amounts of features in the data. Note that applying different thresholds also results in different feature robustness, which may also influence the run times. The applied thresholds and resulting numbers of features are presented in Table 5.2.

5.2.3 Cancer expression data

Data description We employ the dbDEMC 2.0 database (Yang et al., 2017b) to acquire cancer miRNA expression profiles that follow cancer vs control design. We

Data set	α_{bin}	Features
SDF50	0.50	743
SDF45	0.45	1369
SDF40	0.40	2251
SDF35	0.35	3260
SDF30	0.30	4201

TABLE 5.2: Applied α_{bin} and the resulting numbers of features in the reduced simulated data sets. The number in the name of the data set indicates the applied threshold.

search for balanced data sets, i.e., data sets containing an equal number of samples representing both conditions to avoid bias towards one of the classes (Yang et al., 2006; Whalen et al., 2022). Finally, we download three real-world cancer data sets from the NCBI's Gene Expression Omnibus (Edgar, 2002). The details regarding each of the data sets employed in this study are presented in Table 5.3.

GEO Acc. No.	cancer type	samples	positive	negative	miRNAs
GSE22058	HCC	192	96	96	220
GSE10694	HCC	166	78	88	121
GSE36681	NSCLC	206	103	103	1145

TABLE 5.3: Cancer data sets description before pre-processing.

Two hepatocellular carcinoma (HCC) data sets (accessible at NCBI GEO database: GSE10694 and GSE22058, platform GPL10457), as well as the non-small-cell lung carcinoma (NSCLC) data set (accessible at NCBI GEO database: GSE36681), come from matched experiments, i.e., two samples are collected from each subject, one from the cancerous and one from the adjacent non-cancerous site (Li et al., 2008; Burchard et al., 2010; Jang et al., 2012). GSE10694 includes 156 matched samples and ten additional control liver samples. The NSCLC data set consists of two subsets that can be separated by the sample preparation procedure (FF - fresh frozen, FFPE - formalin-fixed, paraffin-embedded). For one of the HCC data sets (GSE10694), we download GEO Series Matrices containing pre-processed data as a non-normalized data matrix is not available. For another HCC (GSE22058) and the NSCLC data set (GSE36681), we download pre-processed data and available non-normalized matrices. All steps of data processing described in this section are recorded and accessible as interactive Jupyter Notebooks for re-use and reproducibility purposes.

Data processing: normalized GEO Series Matrices All the GEO Series Matrices acquired from the database were pre-processed by the authors of the experiments using methods adequate for applied miRNA profiling platforms. Also, the authors normalized all the data using different techniques (Li et al., 2008; Burchard et al., 2010; Jang et al., 2012). Further, we remove ten samples coming from normal liver tissues collected from other subjects (unmatched) from the GSE10694 data set to maintain an equal number of samples in each class. Additionally, for all data sets, we filter out all samples (as well as their pairs, if possible) according to the following normalization quality constraint:

$$M(s_i) \notin [Q_1, Q_3] \tag{5.3}$$

where s_i is a vector of all expression values of sample i, $M(s_i)$ is the median of s_i and Q_1 and Q_3 are the first and third quartile for the whole dataset, respectively. This enables removing extreme outlier samples (those for which the median does not lie between 25% and 75% quantiles for the whole data set). As a result, 14 samples (7 pairs) are removed from the GSE10694 data set. The difference between the data before and after the procedure is shown in Figure 5.3. This procedure does not influence all the other data sets.

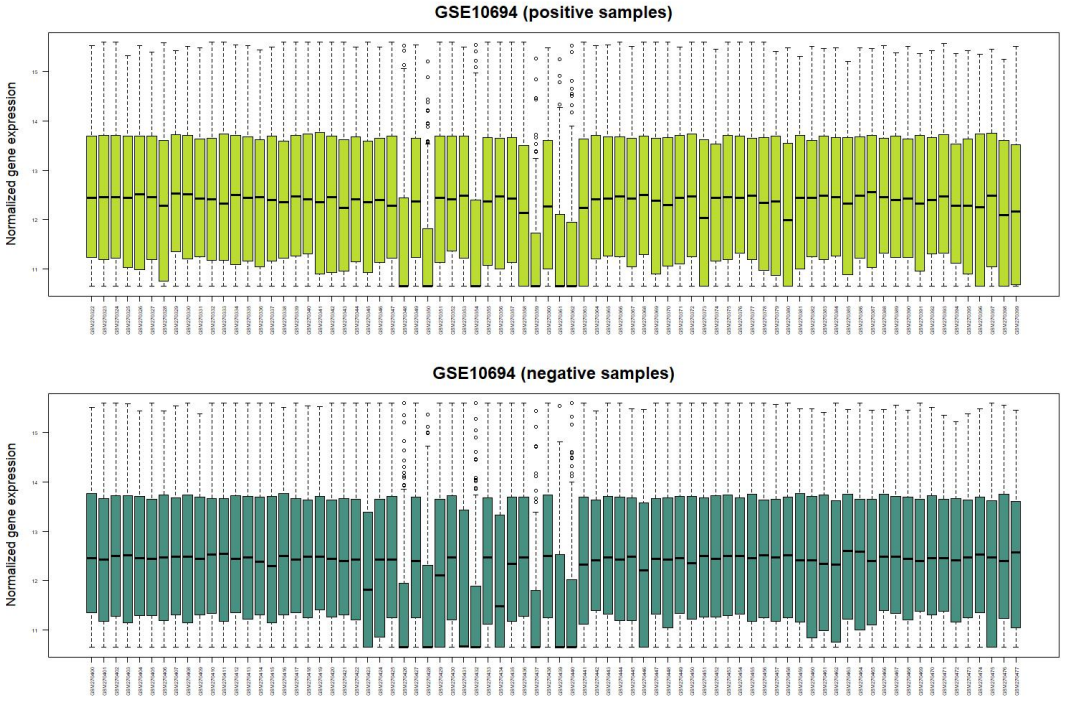
Further, we remove all non-human and precursor (marked with * symbol) miRNAs from each data set as suggested previously by Mohammadi et al., 2017, as precursor miRNAs are not good candidates as circuit inputs. Both HCC data sets (GSE22058 and GSE10694) are divided into five cross-validation folds by subject, i.e., both samples, cancerous and non-cancerous, from the same subject were randomly added to either training or testing data set to avoid information leakage (Whalen et al., 2022). The NSCLC data set (GSE36681) is split into GSE36681-FF and GSE36681-FFPE subsets, according to the applied sample preparation method. Further, both data sets were randomly divided into five folds as the information about sample pairing is unavailable. The data sets were discretized analogously to the procedure described for the synthetic data. The numbers of samples and miRNAs included in the data sets after the processing can be found in Table 5.4. Finally, the data sets are split into five pairs of training and test sets analogously to the simulated data sets.

GEO Acc. No.	cancer type	samples	positive	negative	miRNAs
GSE22058	HCC	192	96	96	210
GSE10694	HCC	142	71	71	109
GSE36681 (FF)	NSCLC	112	56	56	687
GSE36681 (FFPE)	NSCLC	94	47	47	687

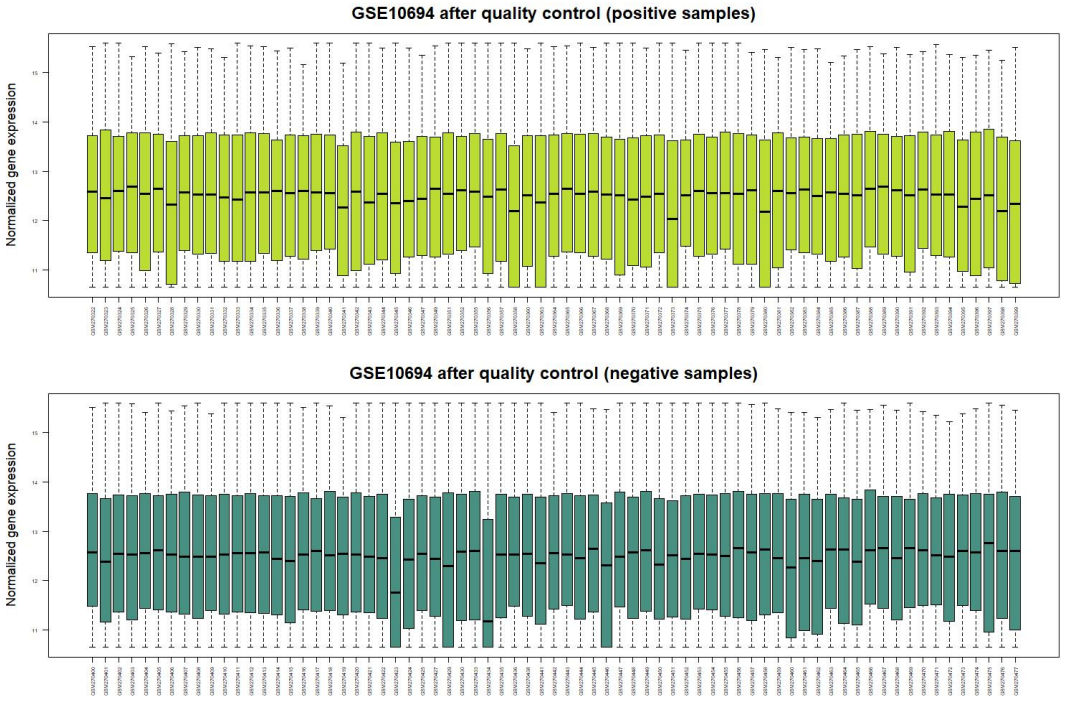
TABLE 5.4: Normalized and non-normalized cancer data sets after pre-processing.

Data processing: non-normalized data For the non-normalized data sets GSE22058, GSE36681-FF and GSE36681-FFPE, we first remove the non-human and precursor miRNAs. Further, the data sets are split into five pairs of training and test subsets. The subsets are next normalized using quantile normalization. To prevent information leakage, similarly to the simulated data described in 5.2.2, we normalize each training and testing fraction separately after the division into cross-validation folds. The number of miRNAs and samples in the data sets are described in Table 5.4. As the normalized and non-normalized data sets comprise identical sets of samples and features, the numbers of samples and features after pre-processing are identical as in the case of jointly normalized data. We will further refer to the non-separately normalized data sets as GSE22058, GSE36681-FF and GSE36681-FFPE and to separately normalized data sets as GSE22058sn, GSE36681-FFsn and GSE36681-FFPEsn.

Discretization All cancer data sets are discretized analogously to the procedure described for synthetic data. Table 5.5 contains the average number of features selected in the process of discretization and the average values of $\Delta_{feature}$ over them, as well as the average values of $\Delta_{feature}$ for all features in the data set before selection. The GSE22058 data set is characterized by higher $\Delta_{feature}$ values compared to the rest of the data sets and the highest number of relevant features. The least relevant features are recovered from the GSE10694 data set, whereas in the case of all



(A) GSE10694 data set before quality control. The y-axis corresponds to the normalized gene expression and the x-axis to particular samples.



(B) GSE10694 data set after quality control. The y-axis corresponds to the normalized gene expression and the x-axis to particular samples.

FIGURE 5.3: GSE10694 data set before quality control (A) and after it (B). The y-axis corresponds to the normalized gene expression and the x-axis to particular samples. Negative and positive sample order corresponds to sample pairs, i.e., the first negative sample is paired with the first positive sample in the plot. 14 samples (7 negative and 7 positive) were removed from the data set.

NCLS data sets, the number of relevant features is similar. The higher the $\overline{\Delta}_{feature}$ is, the higher is also the chance of constructing well-performing classifiers, as more features with high discriminative power are in the data. Thus, we suspect that for data set GSE22058, the performance of the classifiers will be higher than for the rest of the data.

Dataset	Features	$\overline{\Delta}_{features}$ (all)	$\overline{\Delta}_{feature}$
GSE22058	62	0.450	0.632
GSE22058sn	62	0.415	0.631
GSE10694	12	0.313	0.582
GSE36681-FF	22	0.270	0.585
GSE36681-FFsn	24	0.270	0.584
GSE36681-FFPE	21	0.260	0.568
GSE36681-FFPEsn	21	0.252	0.572

TABLE 5.5: The number of relevant features, average values of $\Delta_{feature}$ for all features $\overline{\Delta}_{feature}$ (all) and over differentially regulated features after binarization across the training subsets ($\overline{\Delta}_{feature}$).

5.3 Methods

5.3.1 Algorithm alterations

As the core algorithm, we use the GA architecture applied to the DC problem as described in section 4.2.2. Further, we build on that, focusing on key aspects of the classifier's performance, namely, the objective function and the training procedure. First, we introduce multi-objective optimization of classifiers tailored to the demands of the application, taking into account not only the classifier's accuracy but also single input robustness, as described in the following section. In section 4.2.1, we introduced distributed classifiers in which the clauses (miRNAs) in a single rule could be connected only with an AND operator. Here, we extend the definition of the classifier proposed in section 4.2.1 by allowing an OR operator to connect clauses (miRNAs) in a single rule, creating an OR gate module as proposed in Mohammadi et al., 2017. Furthermore, we elevate the parameter tuning and classifier training procedures to improve the classifiers' performance in terms of generalization facing novel data and to reduce the run-time. We also slightly change the selection procedure by adding randomly chosen best-performing solutions gathered over the past generations in each iteration of the algorithm to the current population. This helps increase the chance of best (also called elite) solutions to participate in a crossover in every iteration of the algorithm and look through the neighbouring points in the search space. Finally, we move our focus to increase the flexibility and accessibility of the tool, as described in the last section.

5.3.2 Objective function

As described in section 4.2.5, we apply balanced accuracy to evaluate the performance of distributed classifiers. However, differentially regulated miRNAs usually diverge in terms of robustness. This can be measured using the global class distribution diversity Δ described in section 5.2.1. The miRNAs that demonstrate high Δ display more consistent discriminative regulation patterns between the classes

across samples. Thus, such miRNAs may be more robust against noise and generalize better to novel information that is not described by the training data. Here, we introduce the second measure of classifier performance DC_Δ calculated according to Eq.5.4:

$$DC_\Delta = \frac{\sum_{i=1}^{n} \Delta_{feature_i}}{n} \tag{5.4}$$

where $i = 1, ..., n$ is the ith miRNA in a finite set $M = \{feature_1, ..., feature_n\}$ and n is the number of unique miRNA inputs in a given DC. DC_Δ describes the average ability of miRNAs in the classifier to distinguish between the two classes. Further, we propose a weighted multi-objective function that allows calculating a combined DC_{score}:

$$DC_{score} = wBACC + (1 - w)DC_\Delta \tag{5.5}$$

where weight $w \in [0, 1]$ determines the importance of each objective, i.e., high w emphasizes the importance of BACC and low w the importance of DC_Δ in the optimization process. The weight allows adapting the significance of both objectives to the data and increases the performance of classifiers in terms of their robustness to noise and novel information. Note that $w = 0$ leads to optimizing only for DC_Δ. Thus, the information about the accuracy of a classifier is not included in the optimization process. Further, $w = 1$ corresponds to the prior objective function proposed in section 4.2.5. For data sets, in which miRNAs significantly differ in terms of robustness (Δ), using DC_Δ allows capturing better candidates for classifier inputs and accelerates the optimization process. Here, one may expect a higher contribution of the DC_Δ to be beneficial. In this study, we apply DC_{score} to optimize classifiers and BACC to evaluate their performance in training and on hold-out test data.

Another important objective in cell classifier design is the simplicity of the circuit. As described in section 4.2.5, the proposed algorithm records all the best solutions found during the GA iterations. After the termination, solutions consisting of the lowest number of inputs (features) are returned as the final best-performing classifiers. Among them, a single solution of top of the list is passed for evaluation.

5.3.3 Training and evaluation scheme

To measure the performance of DCs, we performed tests in a controlled and real-world setting, i.e., on both simulated and cancer data described in Section 5.2. To tune the parameters of the algorithm, train and evaluate the classifiers we employed a nested cross-validation. The outer 5-fold CV is performed for 5 training-validation data pairs resulting from a data division described in Section 5.2. The inner 5-fold cross-validation is used to find the most suitable parameters of the algorithm for a particular training data set. Those parameters are applied to train the classifiers on the training fraction and test them on hold-out test data. The overall workflow is presented in Figure 5.4 and the details are described in further sections. This setup is further applied to all data sets if not specified differently.

Parameter Tuning The training data sets of the outer CV (Figure 5.4) are employed to tune the standard GA parameters: termination criterion (tc), population size (ps), tournament size (ts), crossover (cp) and mutation probability (mp), as well as the multi-objective function weight (w, optional). The objective function weight w can be also fixed to a user-specified value in the process of tuning. Note that balanced

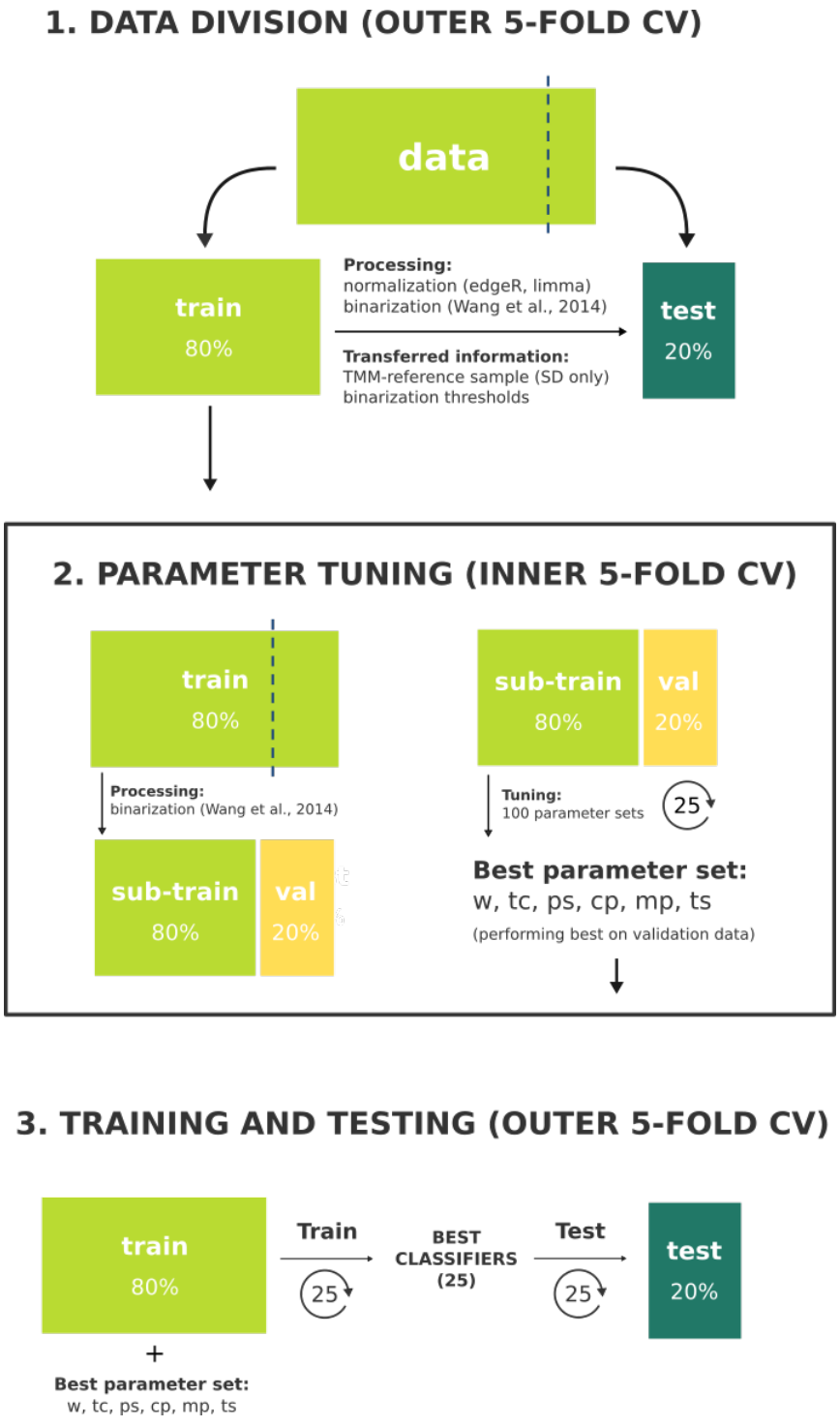


FIGURE 5.4: Training and testing scheme: (1) First the original data set is divided into training and test fractions. Next, the training and testing fractions are separately pre-processed. (2) The normalized training fraction is further used to tune parameters of the GA and the weight w in an inner 5-fold cross-validation. (3) The best parameter set is further applied to train classifiers on pre-processed training fraction and evaluated on the hold-out test data set.

accuracy previously applied in Chapter 4 contributes to the multi-objective function and can be also used as a single-objective function by setting w to 1.0. To increase the performance of the algorithm we refined the termination criterion proposed in 4. Previously, the number of algorithm's iterations was restricted by a pre-defined number of iterations. This can increase the optimization run-time, as well as result in the classifier's under-fitting (Liu et al., 2018). Thus, the termination criterion is specified by the maximal number of consecutive iterations in which the classifier's performance was not improved.

To optimize the parameters we again applied a random search approach (Bergstra and Bengio, 2012). We randomly chose 100 sets of parameters in the following ranges: w: 0.01-1.0, step 0.01, tc: 5-50, step 5, ps: 25-300, step 25, ts: 0.05-0.5, step 0.05, cp: 0.0-1.0, step 0.05, mp: 0.0-1.0, step 0.05.

We performed 5-fold inner cross-validation for each generated parameter set. First, we divided the non-discretized training data into 5 pairs of training and validation fractions and discretized them analogously to the procedure described in Section 5.2. In this step, we omit separate normalization of inner folds. We revisit this issue in section 5.5. For paired data sets (GSE22058 and GSE10694), the samples were split by subject into training and validation fractions. The classifiers were trained on the training data using DC_{score} as the objective function and evaluated on the validation data using BACC. GAs are non-deterministic algorithms and each GA run may result in different results. Each single GA run was repeated 25 times to estimate the performance of the algorithm in a reasonable run time and the average score was recorded for each data fold. The best parameter set was chosen based on the average validation BACC resulting from the 5-fold CV. If two parameter sets resulted in equal BACC, we choose the parameters resulting in lower between-fold standard deviation.

In contrast to the studies presented in Chapter 4, we do not tune the α threshold of the distributed classifiers. Instead, α is here an inherent part of the classifier, affected by the mutation process of the algorithm – changing α to another feasible value is one of the implemented mutation mechanisms. This results in generating classifiers that may differ in the decision threshold but perform equally well in terms of accuracy, providing a variety of feasible architectures as a result of the optimization process. Further, we change the feasible values of α ($\alpha \in 0.25, 0.45, 0.50, 0.75, 1.0$), which allows covering all possible thresholds for all considered sizes, as shown in Table 5.6.

α	1 rule	2 rules	3 rules	4 rules	5 rules
0.25	0	1	1	1	1
0.45	0	1	1	2	2
0.50	1	1	2	2	3
0.75	1	2	2	3	4
1.00	1	2	3	4	5

TABLE 5.6: Thresholds θ in regard to different α ratios and classifier sizes.

Training and testing We applied the chosen parameters to train classifiers on the processed training data sets and evaluated the best-performing models on the processed testing data set. As genetic algorithms are heuristic approaches that do not

guarantee to return identical solutions with every run, we again repeated the training 25 times and recorded average scores for training and test splits.

5.3.4 Implementation note

The data simulation and pre-processing scripts are implemented in R and Python, and are stored in the GitHub repository, as described in the contribution note at the beginning of this chapter. The binarization procedure is implemented in Python based on the algorithm proposed by Wang et al., 2014. The genetic algorithm and a fully automated testing procedure described in the previous section is implemented in Python. One may choose between running a complex analysis corresponding to the described testing scheme or using the algorithm separately to train the classifiers for a given parameter set. The user can set all the parameters employed in this study or use the default configuration.

5.3.5 Automated ASP training procedure

As the manual constraint relaxation impedes the applicability of the ASP-based workflow, I implemented an automated procedure for training and selecting best-performing classifiers called CellClassifierTrainer[1]. I employ a Python wrapper for the Potassco's ASP solver Clingo called clyngor[2]. Clyngor enables embedding the ASP-solving procedure into Python code and processing the solutions in various formats. I embed the workflow described in 3.2.1 (except for the continuous-valued evaluation) and implement the step-wise constraint relaxation procedure. The solver is run separately for each combination of errors (see 3.2.2) between the lower and upper bounds on the number of false positive and false negative errors. Further, among all returned solutions, the classifiers performing with the lowest number of errors in total are selected (see 3.2.2). Then, we process the selected solutions according to their size preserving the shortest classifiers and filter symmetric solutions (see 3.2.3). Given the training and test data sets the solutions are trained according to the above-described procedure and evaluated on the test data (for more information refer to the code repository).

5.4 Results

5.4.1 Simulated data studies

Parameter tuning

Objective function One would expect BACC, used at the same time as the objective function and the evaluation metric, to yield the best classification performance. However, we suspected that our binary classifiers trained on discretized data are prone to overfit the examples covered by the data – in particular, specific sample profiles that can actually be experimental artifacts. By adding the robustness term, we enforce the classifiers to avoid overly relying on any individual examples and generalize better to more consistent patterns in the data. We employ parameter tuning to assess the performance of both objective functions: BACC corresponding to the fitness function previously applied in section 4.2.5 and the multi-objective function DC_{score}. We tune the parameters within the inner 5-fold cross-validation as described

[1] https://github.com/MelaniaNowicka/CellClassifierTrainer
[2] https://github.com/Aluriak/clyngor

in section 5.3.3. We isolate the influence of the objective function on the classifier's performance by substituting solely the functions (the rest of the procedure is identical).

As shown in Table 5.7, the average validation BACC recorded for both objective functions is comparable. For most data sets (except for SDR10), the difference between the scores is below 0.5%. In the case of the DC_{score}, one more parameter must be tuned in comparison to BACC, namely, the weight w. In the tuning, we employ an equal number of parameter sets in both scenarios. This may slightly hinder the performance of the DC_{score} as the search space of parameter sets is larger. Nonetheless, the DC_{score} function performed slightly better on average (the average difference between DC_{score} and $BACC$ is 0.15% in favour of DC_{score}, the largest improvement is 1.05%). Thus, for further training of classifiers, we employ the multi-objective function DC_{score}. Note that $BACC$ is still a part of the DC_{score} function. In the next section, we analyze the weight w tuned for the DC_{score} in the inner cross-validation. The results show the similar performance is not a result of the reduction of the multi-objective function to BACC (see the following section for more details). However, we repeat the comparison of the two objective functions in the cancer data study to investigate the differences between the two objective functions and evaluate their performance against noise and artifacts coming from real-world miRNA expression studies. Interestingly, we obtain better classification performance using the multi-objective function, as described further in section 5.4.2.

Dataset	Objective function	$BACC_{val}$
SDR0	DC_{score}	98.74
	BACC	**98.85**
SDR10	DC_{score}	**98.88**
	BACC	97.83
SDR20	DC_{score}	**96.48**
	BACC	96.16
SDR30	DC_{score}	**94.97**
	BACC	94.95
SDR40	DC_{score}	91.08
	BACC	**91.20**
SDR50	DC_{score}	84.90
	BACC	**84.96**
SDDISP	DC_{score}	**98.75**
	BACC	98.63
SDP50	DC_{score}	99.85
	BACC	**99.86**

TABLE 5.7: Performance of multi-objective and BACC functions in the inner cross-validation. DC_{score} – the multi-objective function balancing BACC and DC_{Δ}, BACC – balanced accuracy, $BACC_{val}$ - average validation BACC resulting from the inner cross-validation.

Weight Further, we analyze parameters resulting from the tuning for the simulated data sets. Figure 5.5 contains box plots of the parameters recorded across five folds of each of data sets. The weights of the DC_{score} (Figure 5.5A) exceed 0.5 for most of the data sets, whereas the medians are below 1.0 for all of the data sets, indicating that the robustness score DC_{Δ} participates in the optimization of the classifiers. The

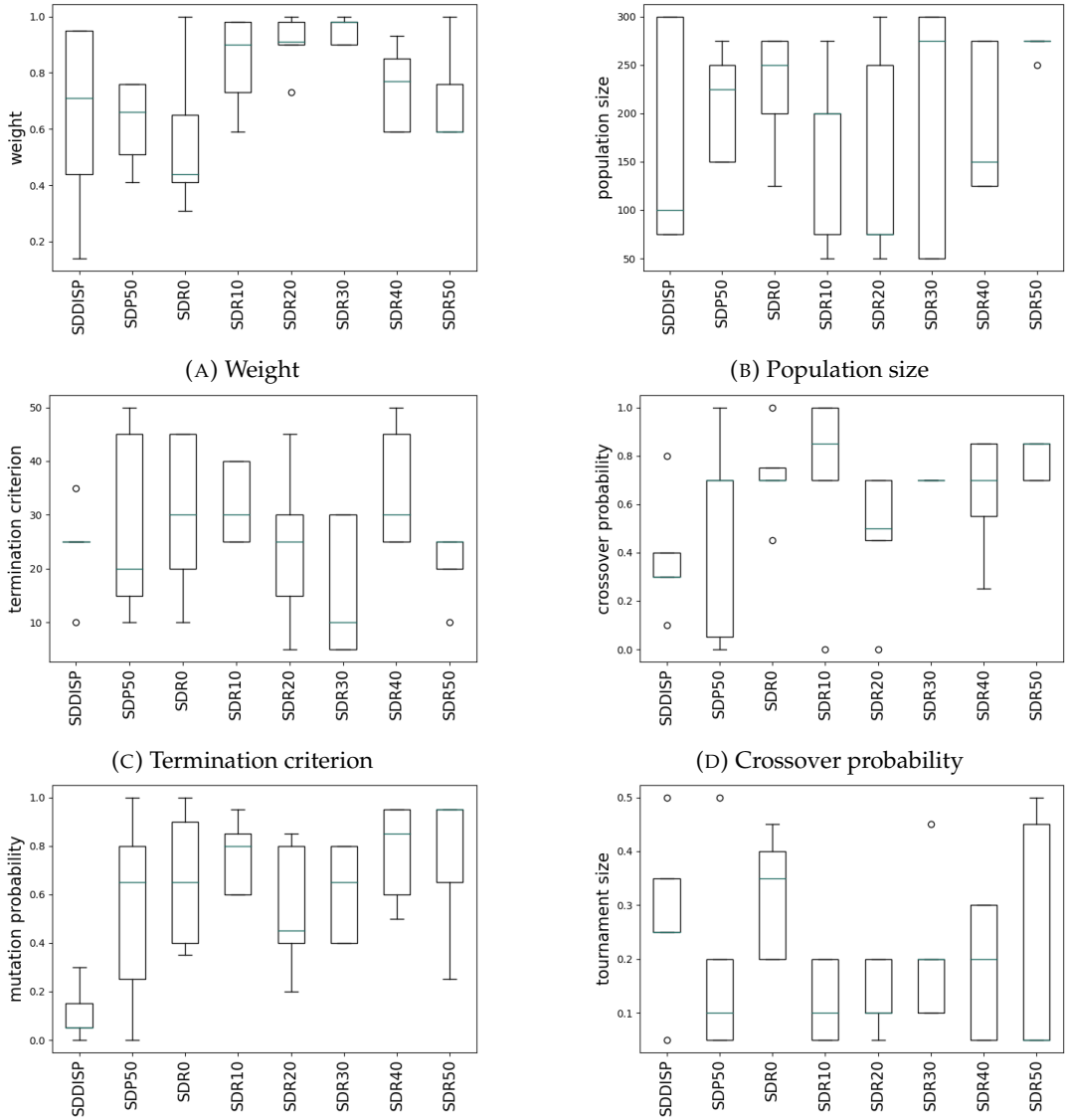
DC_Δ seems to play a more important role for the SDR0 data set (no perturbation), where there can be more good candidate solutions, as well as for the SDR50 data set, where there are many outliers. In the case of different distributions and dispersion, the DC_Δ also seems to influence the search for good solutions.

GA parameters Among the GA parameters, the population size varies fold-wise as well as data set-wise, covering all feasible values. The tournament size is the most steady parameter across the data sets, oscillating around lower values. The crossover probability exceeds 0.5 for most of the data sets. Interestingly, the mutation probability is also relatively high, especially for the data sets containing more outliers. Noisy data sets seem to need more divergence over the populations to achieve better performance. Note that the mutation strategy can alter the classifier's decision threshold. Adjusting the threshold may significantly improve (or impair) the classifier's performance. The termination criterion does not reach the maximum value allowed in the tuning, which allows reducing the algorithm's run-time as the results do not improve over time after reaching a certain point.

Training and performance

We employed the tuned parameters to train and test classifiers within the outer 5-fold cross-validation. The proposed algorithm converges, i.e., achieves high and stable performance over the iterations achieving high scores (average $DC_{score} \approx 96.60\%$, σ of $DC_{score} \approx 3.88\%$). Table 5.8 contains average DC_{score} and average numbers of updates in terms of newly found best solutions over a single training run. As expected, with the rising probability of random outliers in the data set, the achieved DC_{score} decreases. Over a single run, the algorithm finds new best solutions 6 times on average (the average number of new best solution updates for each data set is presented in Table 5.8, Updates). However, the average number of updates differs among data sets. SDP50 (50% Binomial, 50% Poisson) contains several features that perfectly separate both classes (average Δ = 1.00). Thus, it achieves the highest scores and, in fact, does not require further optimization as the best solutions are generated randomly during the population initialization. Nevertheless, over the algorithm's runs, alternative solutions may be gathered resulting in multiple alternative designs. The highest numbers of updates are required for data sets SDR10-40 (random outlier probability of 0.1-0.4). Adding more noise results in a drop of DC_{score} and the number of updates, as the quality of features also severely decreases. At some point (SDR50), the algorithm seems to struggle to find better solutions than the ones found in the first five updates on average.

The algorithm achieves over 95% balanced accuracy in training for all simulated data sets (Table 5.9). With increasing noise (SDR0-50), the training BACC drops and the divergence between folds is more visible (Figure 5.6A). Further, we evaluated the classifiers' performance on hold-out test data fractions. For all data sets, the classifiers' performance is above 89%. The accuracy drops and the standard deviation between folds increases for higher noise levels (between-fold divergence visible also in Figure 5.6B). The accuracy of classifiers is significantly lower for the SDR50 data set, although the median still exceeds 90% (see Figure 5.6). All features that occur in the trained classifiers are truly differentially regulated according to the ground truth. Most of the unregulated features are filtered in the pre-processing step after the discretization, with one exception of the SDDISP data set (different dispersion for each of the conditions; see section 5.2.2 Discretization). However, none of the optimized classifiers consists of features that are not truly regulated.

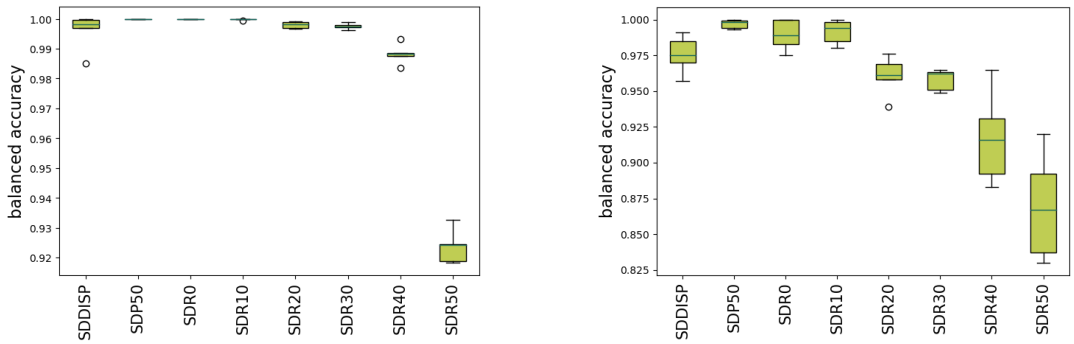


(A) Weight (B) Population size

(C) Termination criterion (D) Crossover probability

(E) Mutation probability (F) Tournament size

FIGURE 5.5: Boxplots representing parameter ranges across five inner cross-validation folds of all simulated data sets.

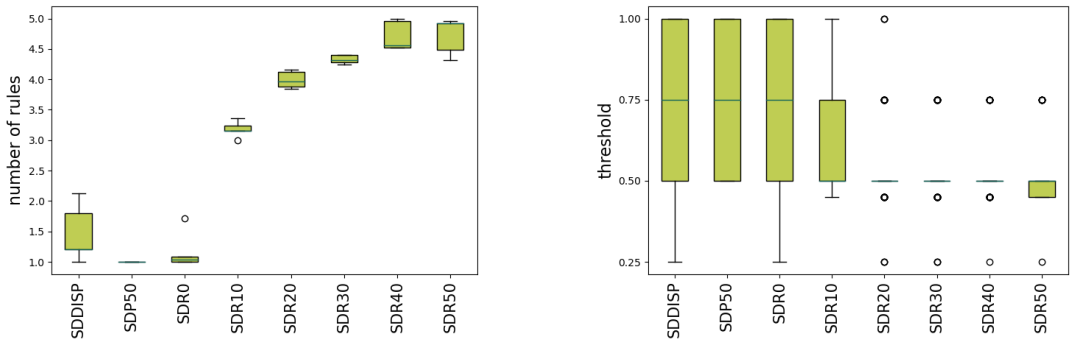


(A) Training BACC (B) Test BACC

FIGURE 5.6: Boxplots representing balanced accuracy across five outer cross-validation folds of all cancer data sets.

TABLE 5.8: Average DC_{score}, DC_{Δ} and number of updates in terms of newly found best solutions in training recorded in the inner 5-fold cross-validation.

Dataset	DC_{score}	DC_{Δ}	Updates
SDR0	99.99	0.99	1.94
SDR10	98.33	0.90	8.54
SDR20	97.86	0.78	9.79
SDR30	98.40	0.70	8.58
SDR40	89.20	0.60	9.12
SDR50	80.70	0.53	4,29
SDDISP	99.37	0.98	3.70
SDP50	100.00	1.00	0.00



(A) Rules (B) Alpha

FIGURE 5.7: Boxplots representing rule numbers and values of the threshold α across five outer cross-validation folds of all simulated data.

Size The number of rules increases with the noise added to the data sets resulting in more complex classifiers (see Table 5.9 and Figure 5.7). For data sets SDDISP, SDP50 and SDR0 the size of a classifier is on average below 2 rules and 2 inputs. The decision threshold is also the most diverse among all the data sets (Figure 5.8). However, the threshold should be interpreted in the context of the classifier's size. For the SDP50 and SDR0, the threshold, in fact, indicates that most of the classifiers are single circuits of maximally 2 inputs. In the case of SDDISP, the classifiers are slightly larger. The size of the classifiers increases with the probability of single outliers in data simulation (Figure 5.7). The median threshold oscillates around 0.5, which means that for most of the classifiers, the decision threshold is similar to a majority vote (where the output is generated if the threshold is reached). This may indicate that the distributed classifiers are more robust to the outliers in the data in contrast to the simple single-circuit classifiers that are prevalent in the case of less noisy data. Here, again using DC_{score} required tuning of one more parameter, which could influence the performance comparison.

Scalability

Here, we focus on measuring the run times of the ASP-based approach and the genetic algorithm. First, we train both methods on SDL (Simulated Data Large) data with a reduced number of features (SDF30-50, see section 5.2.2). We train the ASP

TABLE 5.9: Performance of DCs for simulated data sets: $BACC_{tr}$ - average training BACC, $BACC_{te}$ - average testing BACC, σ_{te} - standard deviation for $BACC_{te}$, $\overline{R}$ - average number of rules, $\overline{I}$ - average number of inputs.

Dataset	$BACC_{tr}$	$BACC_{te}$	σ_{te}	$\overline{R}$	$\overline{I}$
SDR0	100.00	98.94	1.09	1.17	1.18
SDR10	99.99	99.14	0.86	3.18	3.60
SDR20	99.81	96.06	1.40	3.99	5.15
SDR30	99.76	95.80	0.74	4.33	6.22
SDR40	98.82	91.74	3.27	4.71	7.24
SDR50	92.37	86.92	3.50	4.72	6.2
SDDISP	99.60	97.56	1.33	1.46	1.70
SDP50	100.00	99.68	0.31	1.00	1.02

approach using CellClassifierTrainer, described in section 5.3.5. Based on manually performed estimation, we set the upper error boundaries to max 2 false positive and 2 false negative errors. For the genetic algorithm, we use the standard procedure of parameter tuning and training (see 5.3.3). The results are presented in Table 5.10. Although in the case of the ASP-based approach an additional step of parameter optimization is not required, the run-time increases drastically with the additional features. The results could no longer be obtained for the SDF35 data set. In Chapter 3, we have discussed a simulated scalability study for the ASP-based approach (Becker et al., 2018). Here, we employ four times more samples than proposed in Chapter 3 (Becker et al., 2018) in order to explore further potential applications, such as applying our approach to larger gene expression data sets. We also compared the performance of both methods in terms of accuracy. We record the balanced accuracy for training and test fractions for only three data sets using the ASP approach due to run time exceeding the time-out of 900h. The results are presented in Table 5.11. Although both methods achieve very high training BACC (above 98%), the difference between the performance on unseen data is significant. We suspect that the lower performance of ASP is a result of overfitting. Note that GA's run times comprise parameter tuning and training of 25 classifiers. The average time for a single GA run for the feature-reduced SDL data sets (SDF) is 34 sec for the tuned parameters.

TABLE 5.10: Run-time comparison – different number of features.

Dataset	Features	GA run-time [h]	ASP run-time [h]
SDF50	743	12.38	8.15
SDF45	1369	22.82	65.94
SDF40	2251	37.52	853.05
SDF35	3260	54.33	> 900
SDF30	4201	70.02	> 900

Additionally, we measure the run-time and performance of the genetic algorithm employing SDL data set with a decreasing number of samples (SDS24-160). As shown in Table 5.12, the run-time steadily increases with the additional samples. The accuracy recorded in training visibly drops for 24 samples only. Also, the performance on unseen data is significantly lower for the smaller training fractions. This

TABLE 5.11: $BACC_{te}$ recorded for GA and ASP methods on the SDL data sets

Dataset	GA	ASP
SDF50	90.60	83.33
SDF45	89.20	80.00
SDF40	87.60	80.00
SDF35	89.20	-
SDF30	87.10	-

is expected as the least samples are in the training fraction, the more difficult it is for the algorithm to learn the general patterns in the data.

TABLE 5.12: $BACC_{te}$ recorded for GA on the SDS data sets

Dataset	run-time [h]	$BACC_{tr}$	$BACC_{te}$
SDS160	29.05	99.97	87.20
SDS120	23.87	99.60	86.50
SDS80	19.02	99.85	86.40
SDS40	13.80	97.60	70.40
SDS24	11.25	96.83	74.80

5.4.2 Cancer data studies

Parameter tuning

Objective function We repeated the parameter tuning for two objective functions, namely, the multi-objective DC_{score} and BACC. Taking the DC_{Δ} scores into account for most of the cancer data sets improved the performance of classifiers in cross-validation (1.5% on average for all data sets). In the case of the GSE10694 data set, the results are nearly identical. The most significant improvement is recorded for the GSE36681-FFPE data set (3.07%). This suggests that the multi-objective function allows training classifiers to be more robust to various artefacts and sources of noise present in real-world data studies. Thus, we further employ DC_{score} as the objective function in the training of classifiers.

Weight Here, we again explore the parameters resulting from the tuning procedure. Figure 5.8 contains box plots of the parameters recorded across five folds of each cancer data set. The weight of the multi-objective function significantly varies between the data sets. However, most of the values lie between 0.5 and 1.0 (all medians are below 1.0, Figure 5.8A). This indicates that the DC_{Δ} again participates in the optimization of the classifiers, and the DC_{score} allows achieving higher accuracy than BACC alone. For the GSE22058 data set (jointly normalized), the weight is closer to 1.0. We also recorded the highest Δs among the features for this particular data set, suggesting that many are excellent candidates for classifier design (Table 5.14).

GA parameters The best population size seems to be highly dependent on the particular data set. This is also true for the other parameters, although the mutation probability oscillates around higher values. The termination criterion values are

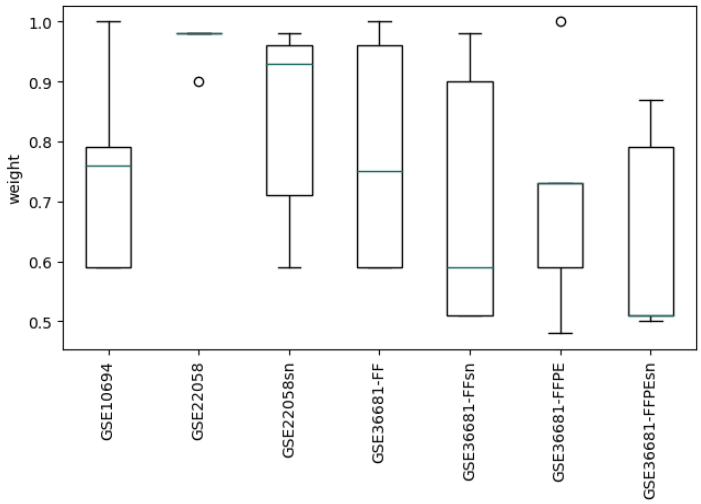
TABLE 5.13: Performance of multi-objective and BACC functions in the inner cross-validation. DC_{score} – the multi-objective function balancing BACC and DC_{Δ}, BACC – balanced accuracy, $BACC_{val}$ - average validation BACC.

Dataset	Objective function	$BACC_{val}$
GSE22058	DC_{score}	**97.51**
	BACC	96.36
GSE22058sn	DC_{score}	**97.93**
	BACC	95.94
GSE10694	DC_{score}	**86.71**
	BACC	86.70
GSE36681-FF	DC_{score}	**82.48**
	BACC	82.18
GSE36681-FFsn	DC_{score}	**84.38**
	BACC	82.63
GSE36681-FFPE	DC_{score}	**88.00**
	BACC	84.93
GSE36681-FFPEsn	DC_{score}	**87.94**
	BACC	85.31

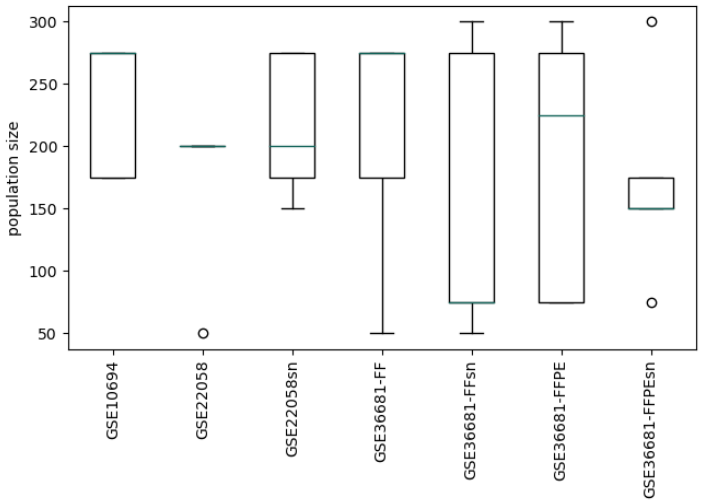
within the set range, suggesting that further iterations would not bring improvement. In contrast to the simulated data, the tournament size varies among almost all possible values between the data sets. Also, the parameters influencing the overall time of computation, e.g., population size or termination criterion, are higher for the largest (in terms of sample number) GSE22058 and GSE10694 data sets, possibly resulting in a higher run-time. Interestingly, all the parameters are very consistent in the case of all folds of the GSE22058 data set, indicating that the folds are highly similar.

Training and performance

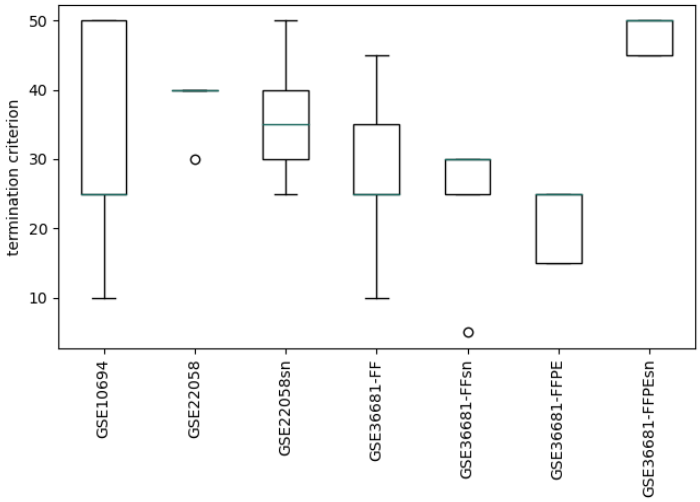
We employ the tuned parameters in training and further evaluate the classifiers in the outer 5-fold cross-validation. The algorithm converges achieving DC_{score} over 90% on average across all data sets ($\sigma \approx 7.02\%$). According to the results presented in Table 5.14, the average between-fold DC_{score} for the GSE22058 data set reaches almost 100%, while the algorithm finds new best solutions over nine times per run on average. For the rest of the data sets, the average DC_{score} and DC_{Δ} are similar. This may indicate that the samples in the GSE22058 data set are fairly straightforward to separate. To explore sample grouping, we visualized the entire non-binarized and binarized data sets with the UMAP dimension reduction technique (McInnes et al., 2020), using scripts I had developed for another study (Bartoszewicz et al., 2022). UMAP is a non-linear dimension reduction technique used for the visualisation of multi-dimensional data in two or three dimensions. UMAP leverages manifold learning techniques to project high-dimensional data into its low-dimensional representation attempting to preserve the local structure of the data (McInnes et al., 2020). Figure 5.10 indicates that the classes are distinguishable in both settings. Interestingly, binarization significantly improves the sample grouping into two classes allowing perfect separation. Note that the binarized data include all samples from the GSE22058 data set and do not indicate the performance of the UMAP model on



(A) Weight

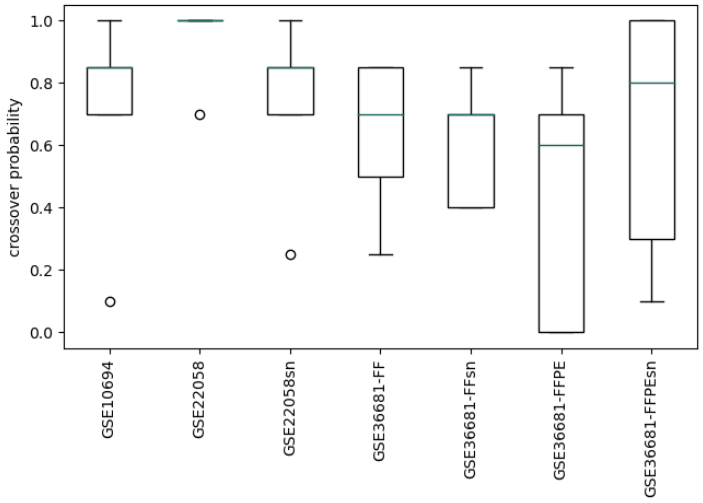


(B) Population size

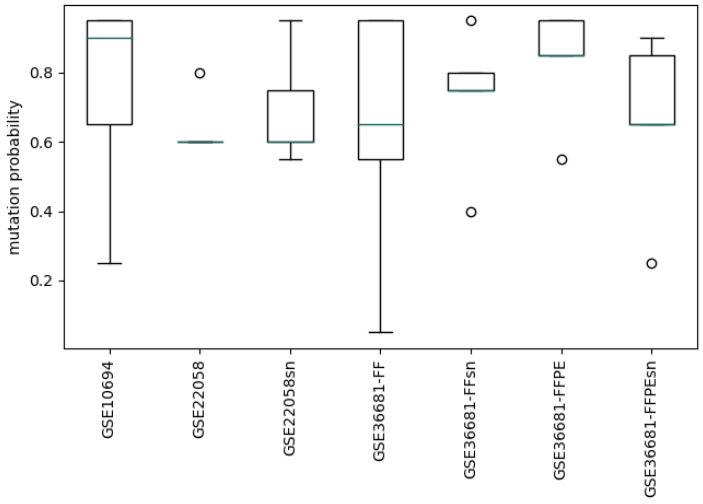


(C) Termination criterion

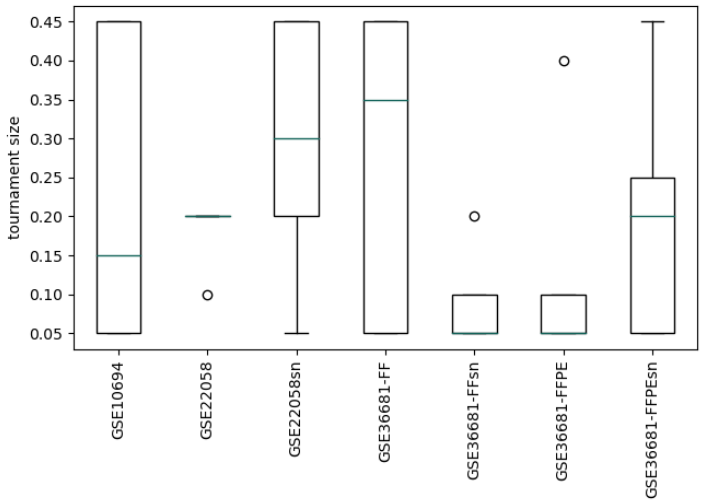
FIGURE 5.8: Boxplots representing parameter ranges across five inner cross-validation folds of all cancer data sets.



(A) Crossover probability

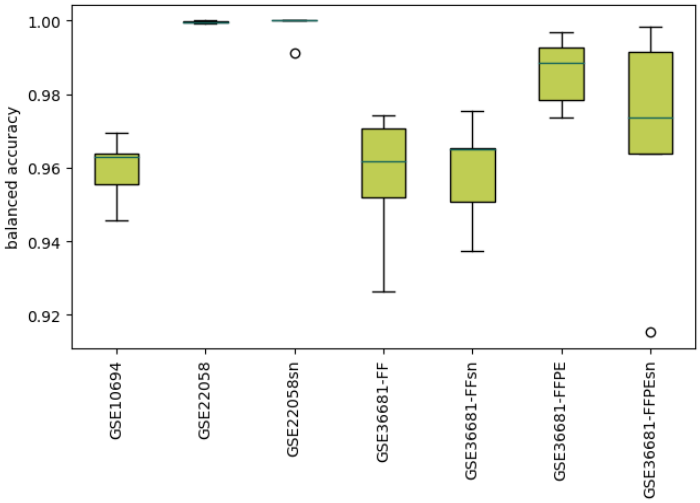


(B) Mutation probability

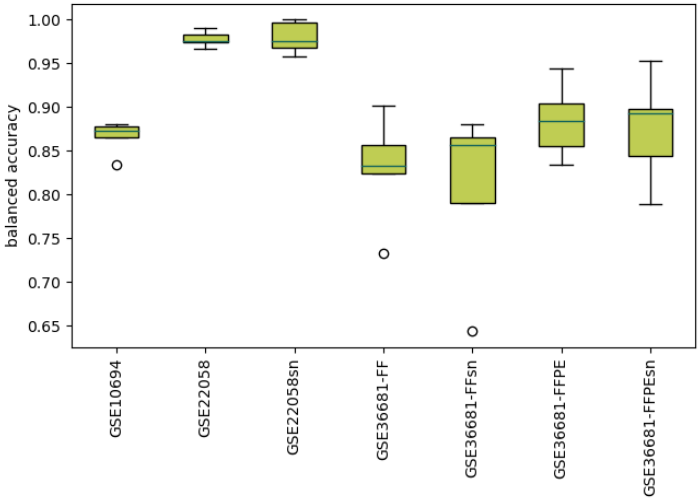


(C) Tournament size

FIGURE 5.8: (continued) Boxplots representing parameter ranges across five inner cross-validation folds of all cancer data sets.



(A) Training BACC



(B) Testing BACC

FIGURE 5.9: Boxplots representing balanced accuracy across five outer cross-validation folds of all cancer data sets.

unseen data. Also, the dimension reduction does not reveal which features are feasible candidates for the classifier nor describe their robustness in terms of sample classification. The high performance of classifiers may also be related to the size of the data set (the GSE22058 data set is the largest of all employed data). The classifiers' accuracy is noticeably lower in terms of validation performance. However, for all data sets, the median is above 80%. Again, the highest performance is recorded for the GSE22058 data sets (jointly and separately normalized). The lowest accuracy can be achieved for the separately normalized GSE36681-FF data set. Similarly, the jointly normalized GSE36681-FF data set does not reach high BACC in terms of the validation data. It may be related to the tissue preparation and preservation technique.

Size The classifiers trained for the cancer data sets are larger in comparison to simulated data sets. The highest median is achieved for the GSE10694 data set. For most data sets, the number of rules is consistent between the folds, except for the separately normalized GSE36681-FF data set. For the same data set, we record the lowest accuracy. For most data sets, the majority vote or thresholds higher than 0.5 are the most suitable.

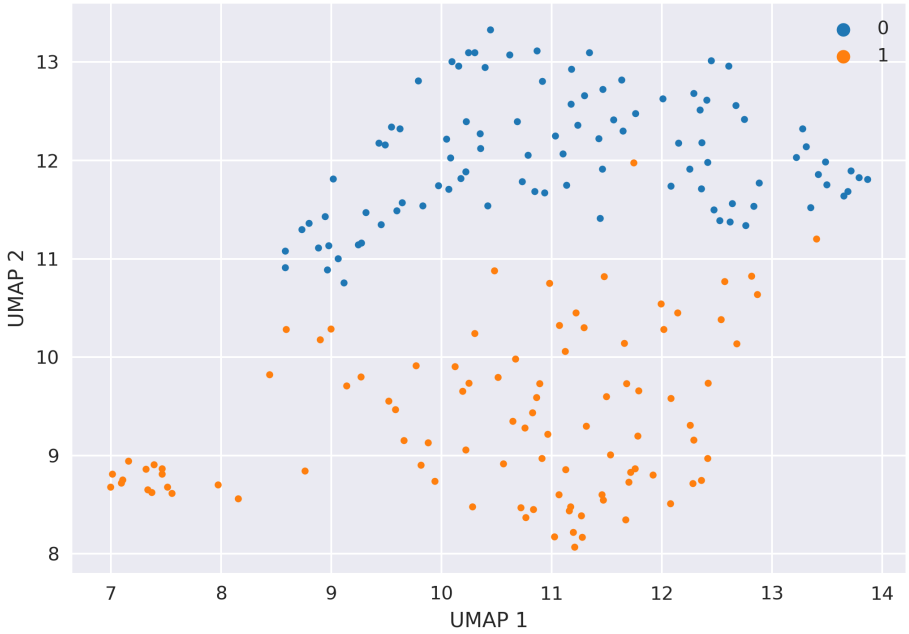
TABLE 5.14: Average DC_{score} and number of updates in terms of newly found best solutions in training recorded in the inner 5-fold cross-validation. sn – separate normalization.

Dataset	DC_{score}	DC_{Δ}	Updates
GSE22058	99.34	0.84	9.59
GSE22058sn	97.14	0.84	9.52
GSE10694	87.00	0.60	6.71
GSE36681-FF	89.03	0.64	6.99
GSE36681-FFsn	86.74	0.66	8.64
GSE36681-FPFE	88.97	0.64	7.00
GSE36681-FFPEsn	87.23	0.68	5.38

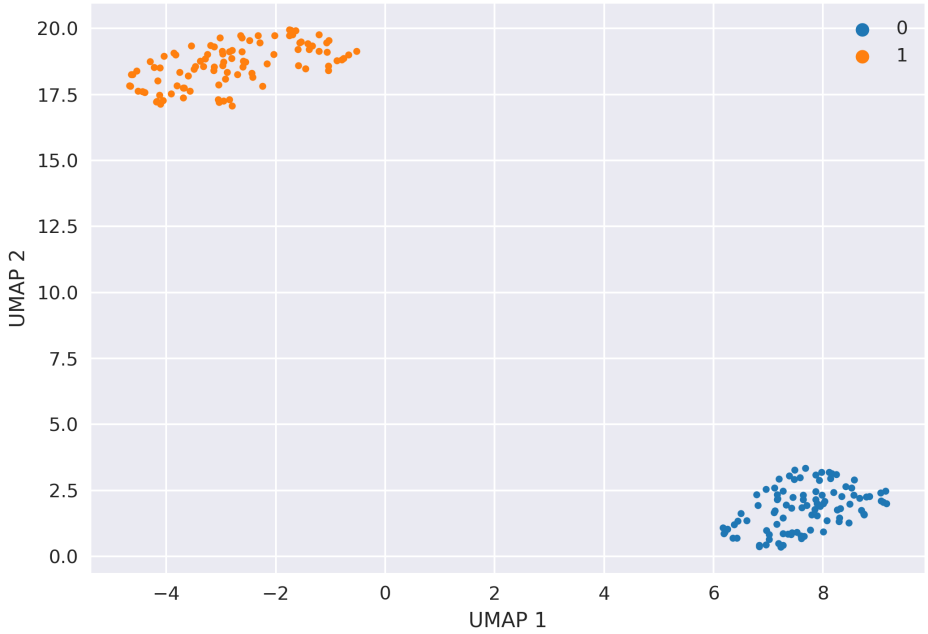
Feature frequencies

In this section, we present exemplary classifiers for different data sets. Also, we analyse frequencies at which different miRNAs occur in the optimized DCs to assess whether the chosen miRNAs are described as differentially regulated in cancer by unrelated studies. Figures 5.12 and S5.4-S5.5 show relative frequencies of miRNAs occurring among all classifiers recorded for each of the data sets.

GSE22058 The most frequent miRNAs occurring in classifiers designed for the GSE22058 data set are hsa-miR-188 (up-regulated) and hsa-miR-450 (down-regulated), presented in Figure 5.12A. We searched the literature and found miR-188 described as down-regulated in HCC tissues in a study by Ma et al., 2019. This is contradictory to our findings. We re-analyzed the original non-normalized and normalized GSE22058 data (Burchard et al., 2010). We performed a rank sum test (similarly to Burchard et al., 2010) on the miR-188 expression profile and found that for this particular data set, miR-188 is indeed up-regulated in the HCC cells belonging to GSE22058 data set (non-normalized data p-value=7.572e-15, normalized data p-value=2.2e-16) and the classifier retrieved a true signal from the data. Another

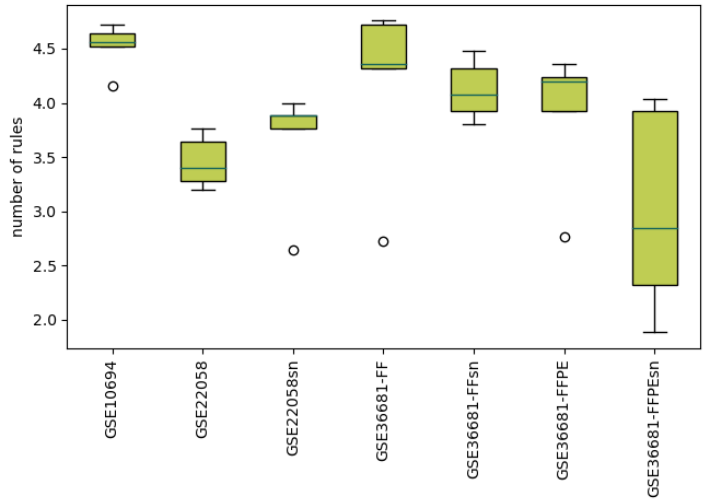


(A) Non-binarized GSE22058 data set.

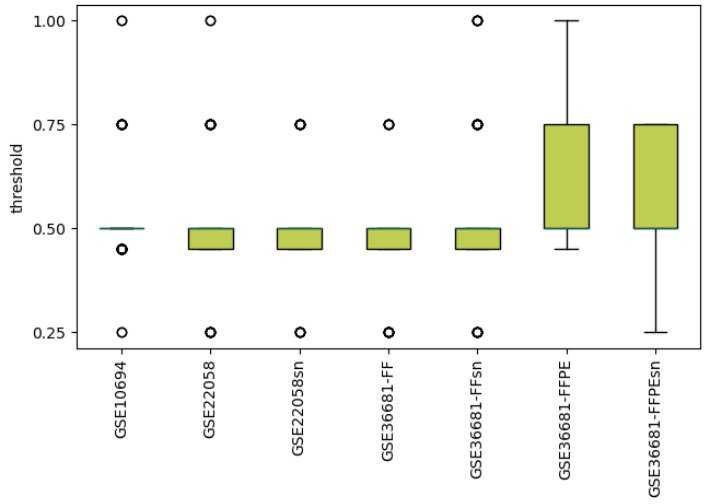


(B) Binarized GSE22058 data set.

FIGURE 5.10: UMAP visualisation of all samples in the binarized and non-binarized GSE22058 data set. Blue/dark (0) - control samples, orange/bright (1) - cancer samples. UMAP1 corresponds to the first dimension and UMAP2 to the second dimension of the low-dimensional projection.



(A) Rules



(B) Alpha

FIGURE 5.11: Boxplots representing rule numbers and values of the threshold α across five outer cross-validation folds of all cancer data sets.

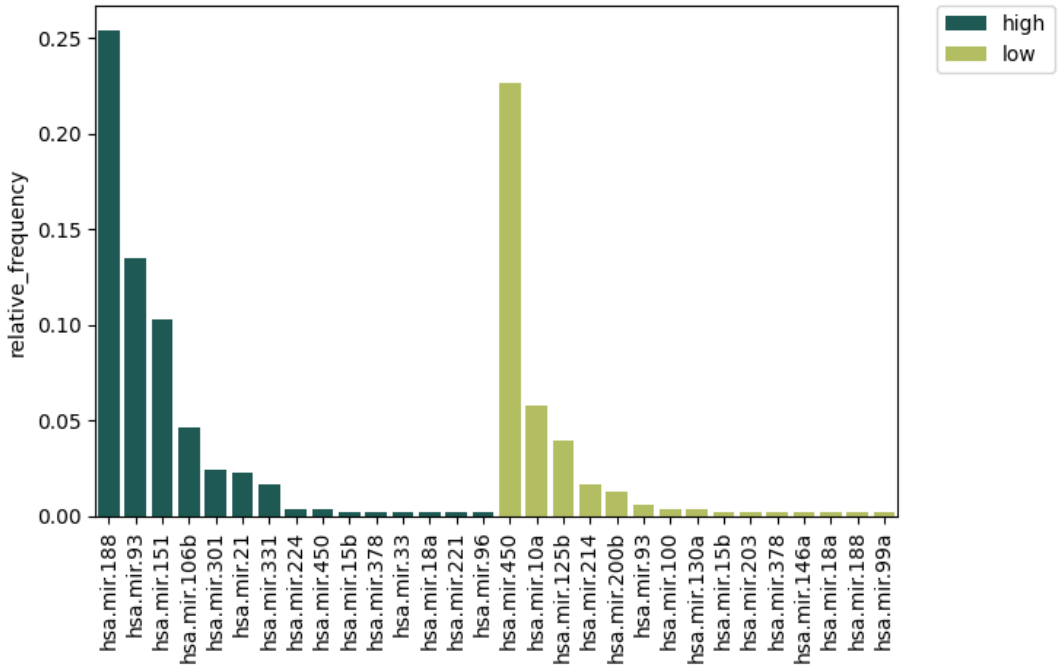
frequent miRNA appearing in the classifiers is down-regulated hsa-miR-450. The miRNA is also described as negatively regulated in the work by Li et al., 2019. The same two miRNAs prevail for both jointly and separately normalized data sets (Figure 5.12. However, the third miRNA is different (hsa-miR-93 for joint normalization, hsa-miR-151 for separate normalization). Strikingly, hsa-miR-93 is rarely used in classifiers trained on separately normalized data. This could be a result of data leakage between the training and validation fractions. Thus, we hypothesize that hsa-miR-151 is more robust while facing more divergence between the training and validation data sets.

GSE10694 In the case of the GSE10694 data set, several miRNAs appeared in classifiers with similar frequency as up-regulated: hsa-miR-221, hsa-miR-222, hsa-miR-224 and hsa-miR-93. Those miRNAs are commonly found as upregulated in HCC patients (Fornari et al., 2008; Pineau et al., 2010; Akkiz, 2014). The most frequent downregulated feature is hsa-miR-422b, previously described in the literature (Akkiz, 2014) as well. Strikingly, there is little overlap among the most frequent features appearing in the classifiers trained for both liver cancer data sets GSE22058 and GSE10694. However, some miRNAs are covered in both cases, e.g., hsa-miR-100 or hsa-miR-93.

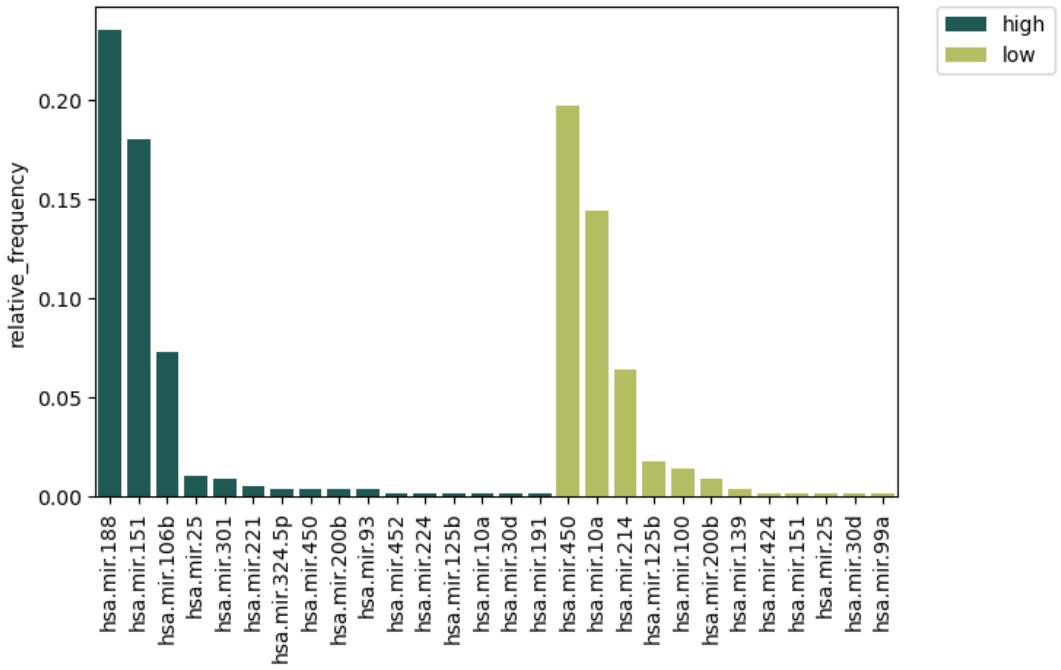
GSE36681-FF and GSE36681-FFPE Figures S5.4 and S5.5 show miRNAs frequencies for jointly and separately normalized GSE36681-FF and -FFPE data sets. One would expect a substantial overlap between the features in classifiers trained on data sets, both coming from NSCLC patients measured using the same platform. However, the sample preparation process seems to have a larger influence on the measurements. The most frequent miRNAs for GSE36681-FF data set are hsa-miR-183, hsa-miR-182 (both upregulated) and hsa-miR-522 (downregulated). hsa-miR-183 and hsa-miR-182 are involved in many different cancers (Suzuki et al., 2018) and increased in case of the NSCLC cancer (Wang et al., 2019). As in the case of the GSE22058 data set, we found a result contradicting our findings in a study by Zhang et al., 2016 regarding the dysregulation of hsa-miR-522. In our classifiers, this miRNA seems to be treated as downregulated. The rank sum test confirms a significant difference between the case and control groups in our data (p-value = 3.424e-08). We comment on this further in section 5.5. In the case of the GSE36681-FFPE data set, the prevalent features are hsa-miR-135b (upregulated) and hsa-miR-218 (downregulated). According to Lin et al., 2013 hsa-miR-135b is upregulated in invasive NSCLC cells. Yang et al., 2017a show that hsa-miR-218 is downregulated in NSCLC cells.

Cross-dataset comparison

Differences in platforms, techniques and processing methods influence the performance of classifiers trained and tested on data sets from different studies. In the cross-dataset comparison, we retrain our classifiers using entire cancer data sets obtained from the same tissues and test them on the unseen data from another study or sample processing technique. This creates two pairs of liver and lung cancer data sets where we repeat the tuning and training on the GSE10694 data set and test on GSE22058, and vice versa (analogously for lung cancer data sets). As the liver cancer data sets do not share the same feature set, we removed all miRNAs that do not occur in the intersection of the two sets. 112 features were removed from the GSE22058 and 11 from the GSE10694 data set. As a results, some of the most frequent features that have previously appeared in the classifiers optimized for the GSE22058 data set



(A) Jointly normalized GSE22058 data set.



(B) Separately normalized GSE22058 data set.

FIGURE 5.12: Frequencies of features in classifiers recorded for (A) jointly and (B) separately normalized GSE22058 data set.

(see 5.4.2) had to be removed, e.g., hsa-miR-188, hsa-miR-151 and hsa-miR-450 (see Figure 5.12). This can make the training of the classifier much more challenging, resulting in worse performance. We again observe high training BACC for all data sets (above 92%). However, the performance on the unseen data is lower than in the case of cross-validation performed within a single data set. This is especially visible for the GSE22058/GSE10694 test, where essential features were removed from the GSE22058, impeding the design of high-quality classifiers. For the GSE36681-FF/GSE36681-FFPE and GSE36681-FFPE/GSE36681-FF data sets, the performance drop is not larger than 4.5%. Interestingly, for the GSE10694/GSE22058, the performance of the classifiers is slightly higher on the GSE22058 data set. This may be caused by the increased number of samples for training or difference in the evaluation schema, i.e., cross-validation vs a single hold-out data test.

TABLE 5.15: Performance of DCs for the cross-dataset comparison: $BACC_{tr}$ - average training BACC, $BACC_{te}$ – average testing BACC, σ_{te} – standard deviation for $BACC_{te}$, $\overline{R}$ – average number of rules, $\overline{I}$ – average number of inputs.

Train data set / Test data set	$BACC_{tr}$	$BACC_{te}$	σ_{te}	$\overline{R}$	$\overline{I}$
GSE10694 / GSE22058	94.62	87.94	0.05	4.48	7.8
GSE22058 / GSE10694	99.88	66.73	0.15	3.72	5.72
GSE36681-FF / GSE36681-FFPE	92.64	78.47	0.01	3.60	5.36
GSE36681-FFPE / GSE36681-FF	98.81	77.25	0.10	4.20	7.08

Single- and multi-circuit comparison

To design single-circuit classifiers we employ the approach proposed by Becker et al., 2018 and the CellClassifierTrainer described in 5.3.5. The method allows optimizing two objectives according to the following hierarchy using the Answer Set Programming solver: (i) the numbers of errors, namely false positives and false negatives and (ii) the simplicity of the classifier. To design single-circuit classifiers, we used the strategy applied for real-world case studies by Becker et al., 2018. We employ parameters that correspond to the classifier constraints proposed by Mohammadi et al., 2017 for all data sets. The method returns all globally optimal solutions that satisfy the pre-defined constraints. We recorded all the best solutions returned for each training data set and evaluated them on the corresponding testing data set using BACC. This allows directly comparing the performance of both classifier designs. In Table 5.16, we present average performance measures obtained for single-circuit and distributed classifiers.

Distributed classifiers achieve higher accuracy in terms of training for two data sets (GSE10694 and GSE36682-FFsn). For the remaining data, the results are comparable for both approaches. However, when facing the unseen data, the multi-circuit classifiers outperform the single-circuit designs by up to 13.40%. This indicates that the distributed classifiers generalize better to novel data. Also, the high performance of single-circuit classifiers may be an artefact of information leakage between the data sets. DCs bring improvement in both sensitivity and specificity for most of the cases (except for GSE36681-FF and GSE36681-FFsn data sets). Although the results for jointly normalized GSE36681 data sets are comparable (with ca. 1% improvement for single circuit designs), the separately normalized data shows that the single-circuit classifiers are also much more sensitive to divergences between the

TABLE 5.16: Performance measures for single-circuit (SC) and distributed classifiers (DC) for four cancer data sets: $BACC_{tr}$ – average training BACC, $BACC_{te}$ – average testing BACC, σ_{te} – standard deviation for $BACC_{te}$, TPR – True Positive Rate, TNR – True Negative Rate, $\overline{R}$ – average number of rules, $\overline{I}$ – average number of inputs.

Dataset	Method	$BACC_{tr}$	$BACC_{te}$	σ_{te}	TPR	TNR	$\overline{R}$	$\overline{I}$
GSE22058	DC	99.96	**97.78**	0.91	96.85	98.70	3.46	4.28
	SC	100.00	95.69	1.80	94.71	96.67	-	4.60
		-0.04	2.09	-0.89	2.14	2.03	-	-0.32
GSE22058-sn	DC	99.82	**97.94**	1.84	98.23	97.64	3.63	4.38
	SC	100.00	93.23	3.52	91.84	94.63	-	5.00
		-0.18	4.70	-1.68	6.39	3.02		-0.62
GSE10694	DC	95.94	**86.60**	1.86	84.32	88.89	4.52	7.62
	SC	92.96	80.00	4.86	77.71	82.29	-	5.00
		2.98	6.60	-3.01	6.61	6.60		2.62
GSE36681-FF	DC	95.70	82.96	6.17	80.64	85.28	4.18	6.94
	SC	95.10	**84.28**	6.61	86.44	82.12	-	4.00
		0.60	-1.32	-0.43	-5.80	3.16		2.94
GSE36681-FFsn	DC	95.87	**80.71**	9.77	75.07	86.35	4.12	6.26
	SC	94.87	76.25	5.09	72.73	79.77	-	4.40
		1.00	4.46	4.68	2.34	6.58		1.86
GSE36681-FFPE	DC	98.60	88.47	4.28	87.94	88.99	3.90	5.14
	SC	95.48	**89.78**	5.60	89.36	90.20	-	5.20
		3.11	-1.31	-1.32	-1.42	-1.21		-0.06
GSE36681-FFPEsn	DC	96.85	**87.56**	6.19	85.00	90.12	3.00	4.30
	SC	96.29	74.16	14.44	70.89	77.44	-	4.60
		0.56	13.40	-8.26	14.12	12.68		-0.30

training and test data. Robustness to novel and noisy data is crucial for the therapy to be effective and non-toxic to patients. In terms of the classifier size, the multi-circuits consist of a larger number of inputs on average compared to single circuits. This may pose new challenges to the application of distributed classifiers in practice. We discuss the advantages and drawbacks of both approaches in section 5.5.

5.5 Discussion

In Chapter 4, we have shown the potential of multi-circuit classifiers to perform with higher accuracy than their single-circuit counterparts. However, the employed evaluation strategy provides narrow insight into assessing the classifier's performance in practical scenarios. In this chapter, we address the caveats of the previously presented study design and focus on refining the training and evaluation strategies to increase the robustness of multi-circuit designs in the context of cancer cell classification. In particular, we revisit the classifier's optimality criteria and focus on building a comprehensive scheme that enables a thorough estimation of the generalization error of the circuits.

We developed a multi-objective function that allows combining two optimization criteria, the accuracy of the classifier and the robustness of employed miRNAs. The importance of particular criteria may be defined by a tunable weight. We assessed the influence of the function on the generalization error by comparing the combined objectives and balanced accuracy using simulated and cancer data. Although in the case of synthetic data, including the robustness score in the optimization process did not improve the classifier's performance, the multi-objective function enabled reduction of the generalization error for cancer data. The weight seems to be data-dependent, and thus, it is recommended to tune the weight for a particular data set.

To measure the classifier's performance in a controllable environment, we perform several simulated and cancer data studies to measure how well multi-circuit classifiers recognize novel samples. First, we generate synthetic data sets with different intensities of data perturbations. As expected, the prediction accuracy of classifiers decreases with the increase in the data noise. However, the multi-circuit designs demonstrate resistance to such perturbations as the validation accuracy does not drop below 85% for all simulated case studies. Also, the classifiers easily capture the ground truth signal in the data. We also utilize the simulated data to perform a scalability test for both the ASP-based and GA-based computational frameworks and show that GA is more efficient in terms of computation time for large data sets.

Further, we employ multiple cancer data sets to estimate the generalization error in a real-world scenario. The cellular environment the circuits encounter in real-world scenarios is in practice only partially described by the data. Thus, the classifiers must be particularly resistant to noise. In this chapter, we emphasize the importance of appropriate data preprocessing in the training of classifiers. In particular, we show the influence of joint preprocessing of training and validation data on estimating generalization error. We perform cancer case studies employing two data preprocessing setups - we normalize the data sets before and after the division into training and validation subsets. Also, as we employ supervised discretization of data, to avoid information leakage between the training and validation data but preserve the existing signal, we discretize the data separately. In the case of the single-circuit designs, the performance substantially drops for all the data sets. This may suggest that the performance estimated based on jointly normalized data sets is overly optimistic and is a result of information leakage between the data. The results show that multi-circuit designs are more robust to the larger divergence between the training and validation data as the results for jointly and separately normalized data sets are comparable. We further discuss the reasons underlying the better performance of multi-circuit classifiers in chapter 7

Chapter 6

Further applications and extensions

Contribution note

Pascal Iversen, Bernhard Y. Renard and Katharina Baum conducted a study on the network- and machine-learning-based approach prediction of drug response using Genomics of Drug Sensitivity in Cancer (GDSC) data, which is not a part of this thesis. Seeing their results, MN had the initial idea that the curated datasets could be employed to assess the performance of the approach presented in Chapter 5 beyond the context of cancer classifier circuit design. MN downloaded the GDSC data on 18.05.2023. PI provided scripts that were previously used in the original study for initial data preprocessing, including matching the GDSC profiles with their IDs, formatting and removal of rarely occurring mutations[a]. MN executed PI's scripts on the data downloaded on 18.05.2023, preprocessed and formatted the outputs of the scripts for compatibility with own approach, trained the presented classifiers and performed the analysis as described in this chapter.

The phage lifecycle prediction study was performed jointly with Jakub M. Bartoszewicz. Both authors contributed equally to this work. MN and JMB conceived the study. MN collected and preprocessed the genome data. JMB trained the neural network and extracted phage features. MN designed the experiments on training logic-based classifiers, prepared the necessary configuration and supervised JMB as he executed the training script according to her specification. MN analyzed the results. MN described the study design and analyzed the presented results.

[a] https://github.com/PascalIversen/LogicGDSCUtils

6.1 Introduction

In the previous chapters, I focused on a particular application, namely, cell classifier circuits in the context of cancer diagnosis and treatment. However, the proposed methods may be applied beyond this scenario, particularly in the context of other binary classification problems. In Chapter 5, I presented synthetic data studies that use a gene expression data simulator and indicate that our classifiers perform well not only in the context of miRNA expression data. Here, I explore alternative classification problems to highlight the flexibility of the proposed workflows in terms of the employed data. In particular, in the first case study, I employ multimodal data, i.e., data consisting of distinct modalities such as different molecular features to predict cancer drug response in cell lines unseen in training. I show that our workflow

enables optimization of classifiers integrating different modalities, such as gene expression and mutations. Further, in the phage lifecycle prediction study, I employ data outside the cancer context to demonstrate the performance of our approach in terms of distinct classification problems. Finally, I discuss additional features and the future outlook of the proposed methods.

6.2 Multimodal data: predicting drug response

In this section, I show that our workflow can be applied to multimodal data. As an example of such a use-case scenario, I focus on the binary classification of drug response represented as half maximal inhibitory concentration (IC50). IC50 measures the amount of a particular drug needed to *in vitro*-inhibit a given biological component or process by 50%. This study focuses on predicting IC50 as high or low for erlotinib, a drug used in treating non-small cell lung and esophageal cancer (Mao et al., 2022). We focus on predicting the resistance of erlotinib based on gene expression and mutation data. The higher the IC50 of a given drug is, the more compound is needed to trigger the desired effect. This results in higher resistance of the targeted cells and amplifies the drug's toxicity due to increased dosage.

We employ gene expression and mutation data to optimize classifiers consisting of multimodal features to predict whether the IC50 for erlotinib is low or high in new cell lines. This can be achieved by coupling both modalities into a unified discrete data set and joint optimization of classifiers comprising diverse features. Figure 6.1 presents an overview of the study.

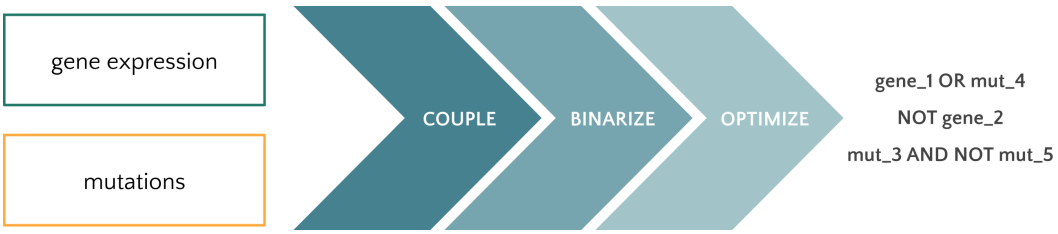


FIGURE 6.1: Overview of the drug response prediction study. Gene expression and mutation data are coupled, and the non-binary features are discretized. Using the joint data set, the classifiers comprising multimodal features are optimized.

The data was acquired from the Genomics of Drug Sensitivity in Cancer (GDSC) database that gathers gene expression, mutational, methylation and other data for 1000 human cancer cell lines and hundreds of compounds, including erlotinib (Yang et al., 2012). I downloaded gene expression and mutation data for all cell lines and compounds (accession date: 18.05.2023) as well as the corresponding IC50 values (given as $ln(IC50)$). The data were preprocessed and formatted using scripts provided by Pascal Iversen, including mapping the IDs to the corresponding features. In the case of mutation data, the rarely mutated features (altered in less than five cell lines) were removed from the data. I further preprocessed and formatted the data to make them compatible with our workflow (code available in `https://github.com/MelaniaNowicka/RAccoon`). First, I filtered gene expression

and mutation cell line profiles that match with erlotinib response and find 374 profiles consisting of 16251 gene expression and 16251 mutation profiles. After concatenating the gene expression and mutation profiles, the final data set comprises 374 annotated samples and 32502 features.

Our workflow requires binary annotation of samples. We visualize the frequency of $ln(IC50)$ values (erlotinib response) for 374 cell lines in a histogram presented in Figure 6.2. We discretize the $ln(IC50)$ values into two levels, low and high, according to two strategies: (1) using a single threshold – median, where all values of $ln(IC50)$ below the median are 0 and above or equal to the median are 1, (2) using two thresholds – first quartile (Q1) and third quartile (Q3), where all values of $ln(IC50)$ below or equal to Q1 are 0, and above or equal to Q3 are 1. The median threshold does not seem to separate the classes well, as many samples yield IC50 values close to the median. This can pose a major challenge for the classifier design. Q1 and Q3 separate the samples with a larger margin. However, employing Q1 and Q3 as thresholds results in removing half of the samples in the data set. I compare the classifier's performance in both setups.

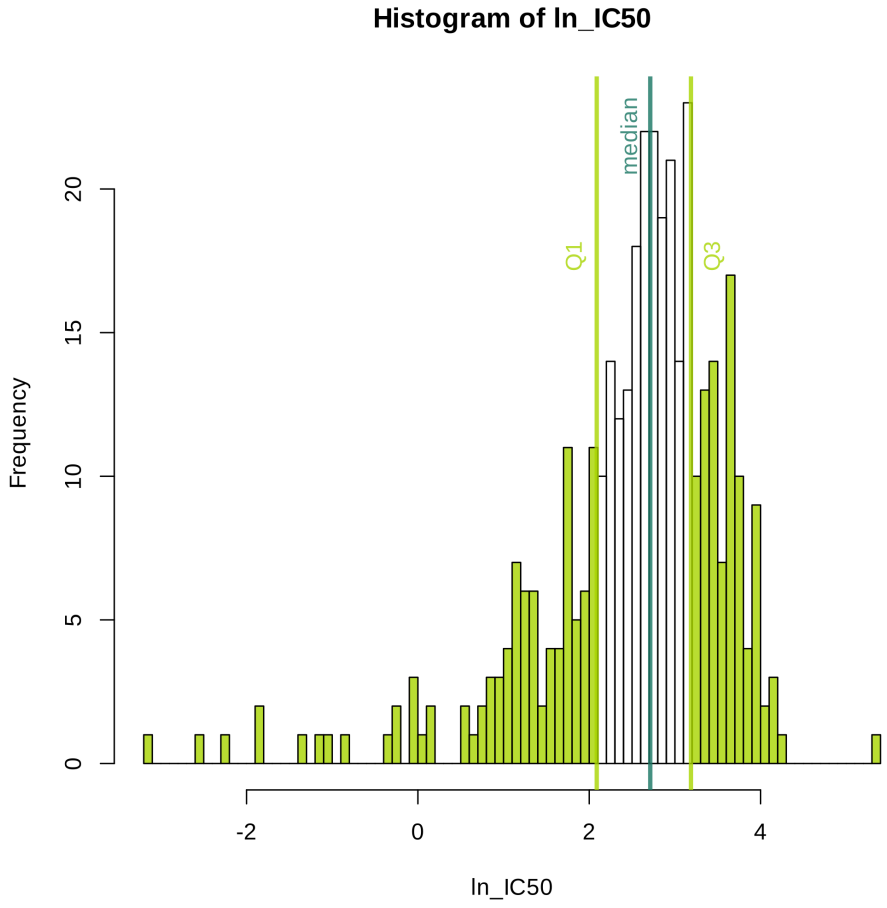


FIGURE 6.2: A histogram of $ln(IC50)$ values for erlotinib. The green bars correspond to negative and positive class samples as discretized using Q1 and Q3 as thresholds.

Further, I perform a nested 5-fold cross-validation as described in the previous chapter (see section 5.3.3). As the mutation data is discrete (it records whether a

mutation occurred in a gene or not), I binarize only gene expression data according to the procedure described in section 5.2.1. The concatenated data containing non-binary and binary features can easily be used as input to the workflow (without the need to handle the binarization steps separately), as binary features are simply omitted in the discretization process. The average between-fold $\Delta_{feature}$ of the two-state gene expression features is 0.22, indicating that many genes do not separate classes well. This may result in large classifier designs and lower performance. Also, to discretize the gene expression, I use α_{bin}=0.2, as applying the default threshold (0.5) results in the discretization of all features into one state. Nonetheless, the vast majority of gene expression features are removed from the data. After removing non-relevant features, the data folds consist of 16228 features on average.

As most of the features in the data lack the $\Delta_{feature}$ score (the majority of the features in the joint data set belong to mutational data), we do not employ the DC_{Δ} in the optimization process (w=1.0). This results in relying solely on BACC as the objective function. Except for the weight, we perform the parameter tuning analogous to the description provided in section 5.3.3. If not stated otherwise, all other parameters are set to the default parameters proposed in Chapter 5. In this study, I reduce the number of GA run repeats to 10 to decrease the run time due to a large number of features.

For the median-based discretization, the classifiers achieve balanced accuracy ($BACC$) of 72.22% in training and only 53.59% on the test data. The low training accuracy indicates underfitting. In addition to that, the classifiers generalize poorly to novel cell lines. In comparison, the training accuracy (82.23%) is ten percentage points higher in the case of quantile-based discretization. However, the performance on the test data increased only by 4 p.p. (58.26%). Note that we use the default classifier size (maximally 10 features). The results may be potentially improved by extending the maximal size of classifiers.

In both cases, the classifiers consist of 5 rules and almost 10 features on average (out of around 16000 possible features in the data set), reaching the maximum allowed size of classifiers. This indicates that extending the classifier's size could increase the performance. As expected, the optimized classifiers comprise various genes and mutations. The most frequently occurring gene in the classifiers is PCLO, which is used as a positive feature (highly expressed). PCLO gene was found up-regulated in esophageal squamous cell carcinoma, and the overexpression of the product protein Piccolo is correlated with poor prognosis (Zhang et al., 2017). The gene is also frequently altered in many different cancers (Zhang et al., 2017). Although the performance in terms of accuracy is unsatisfactory, the classifiers are inherently interpretable and may uncover features of interest. We revisit the potential future directions in the discussion. The run time (including parameter tuning and the final training) is around 23 hours for the quartile-binarized annotation data and around 42 hours for the median-binarized annotation data. The computation was performed on 25 AMD EPYC-Rome cores.

6.3 Beyond cancer: phage lifecycle prediction

This section presents an example of using the approach proposed in Chapter 5 beyond cancer research. In particular, we focus on increasing the interpretability of deep learning models by uncovering interesting features. We present an exemplary study in which we apply our workflow to predict the life cycle of bacteriophages – bacteria-infecting viruses.

Rapid development of antibiotic-resistant bacteria endangers the efficacy of currently available antibiotics (Thompson, 2022). According to WHO, antibiotic resistance poses one of the major threats to global health worldwide, and it's estimated to cause more than 10 million deaths per year by 2050 (Thompson, 2022). The design of new antibiotics, although possible, is time-consuming and expensive (Spellberg, 2014). However, there are existing alternative therapeutics that are yet underexplored. An example is phage therapy, which employs bacteriophages (phages), natural enemies of bacteria, to treat bacterial infections (Hibstu et al., 2022). However, to be efficient, therapeutic phages must fulfil certain criteria. Phages are host-specific, i.e. they infect specific species and strains of bacteria. Besides that, effective therapeutic phages follow a so-called virulent life cycle (Kortright et al., 2019). After infecting the target bacteria, virulent phages replicate, resulting in the host's death. In contrast, temperate phages incorporate their genetic material into the chromosome of the host and do not cause immediate harm to bacteria. Thus, the virulent life cycle is crucial for the therapy to be effective. In this study, we focus on predicting the phage's life cycle directly from the DNA sequence and uncovering a shortlist of features that are responsible for virulent phage behaviour.

We downloaded 3186 publicly available phage genomes from PhagesDB (Russell and Hatfull, 2017) with life cycle annotation (accessed on 27.04.2021). In the case of life cycle prediction, temperate phages belong to the negative class (0) and virulent phages to the positive class (1). We split the data into training (80%), validation (10%) and test (10%) fractions in a balanced manner, i.e., both classes are equally represented in the data fractions.

Next, we follow an established protocol from Bartoszewicz et al., 2021b to train a deep learning model (belonging to the DeePaC package described in Bartoszewicz et al., 2020) for the prediction of the phage life cycle. Briefly, we simulate sequencing reads for each genome and train a deep learning model to distinguish reads belonging to either negative or positive class (for more details, see Bartoszewicz et al., 2021b, where a similar approach was used to predict whether a virus infects humans). To predict the life cycle associated with a particular genome, we average predictions for reads generated from this genome as described in Bartoszewicz et al., 2021b.

We employ the above-described model to extract the 512 real-valued activations of the model's penultimate layer for each phage (further referred to as features) learned to distinguish between the classes (see Bartoszewicz et al., 2022 for more details, where a similar procedure was used in the context of fungal pathogenicity prediction). Briefly, the features correspond to the outputs of the global average pooling layer of the residual network consisting of 18 layers. The detailed architecture of the network is described in Bartoszewicz et al., 2021a. All activation values are non-negative. We format the data according to the description provided in section 2.3, concatenating the phage life cycle annotation with the feature profiles. The final data set comprises phage IDs, life cycle annotation and feature values of 2548 training, 319 validation and 319 test samples.

We employ the above-described data and run a DC_{score}-based optimization of classifiers as described in the previous chapter. If not stated otherwise, all parameters are set to the default proposed in Chapter 5. We reduced the number of GA run repeats to 10 to decrease the run time. In the process of discretization, 258 features were removed as non-relevant ($\overline{\Delta}$ = 0.72). Although the $\overline{\Delta}_{feature}$ is high, the multi-objective function weight chosen in the tuning (w=1.0) indicates that the optimization of classifiers was performed solely using BACC. Strikingly, although our classifiers consist on average of only 5.6 features, the performance ($BACC_{tr}$ = 99.99%,

$BACC_{te}$ = 98.72%) is very similar to the performance achieved by the DeePaC model ($BACC_{te}$ = 99.06%). Thus, we can use the proposed method as a feature extraction approach and build simpler classifiers. Figure 6.3 presents the frequencies of features occurring in the optimized classifiers. The most frequently occurring component is feature 200, appearing as negative. Such features can be used to measure their enrichment in the bacteriophage genomes and identify life cycle-related genes as previously done in Bartoszewicz et al., 2021b. The computation time for 258 features and 2548 training samples (including parameter tuning and the final training) is around 3 hours and 52 minutes. The computation was performed on one AMD EPYC-Rome core.

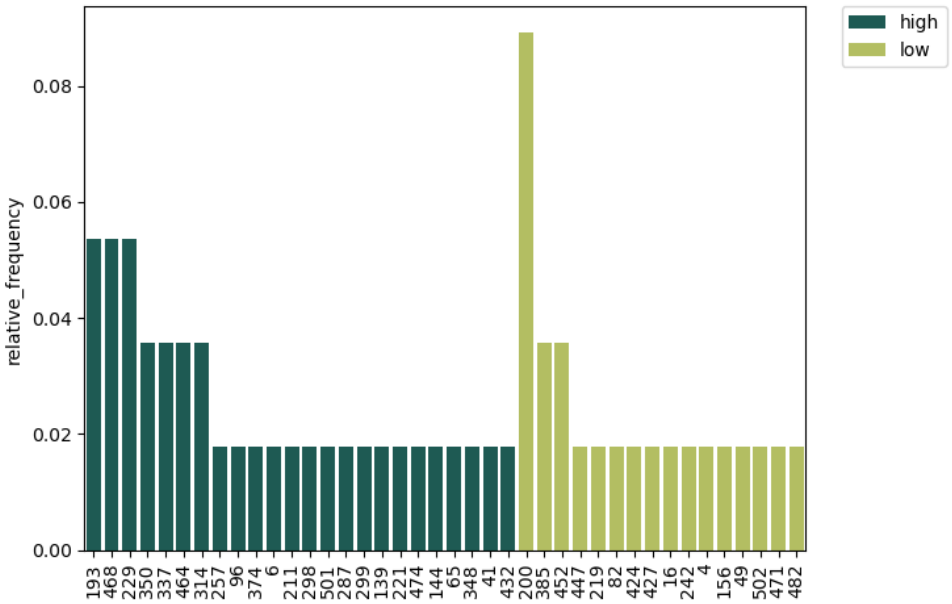


FIGURE 6.3

6.4 Discussion

In this chapter, I briefly showcase alternative applications of the workflow within and beyond cancer research. First, I present a case study on the prediction of binary drug response using multimodal data comprising gene expression and mutational information. In the second study, I describe an exemplary study on predicting bacteriophage life cycle, employing the proposed GA algorithm to compress large classifiers while maintaining high accuracy.

Even if reduced to a binary response, drug response prediction poses a challenge to the research community (Parca et al., 2019; Schätzle et al., 2020). Generalization to novel cell lines is a difficult task, potentially due to the cell heterogeneity (see Chapter 1). In the presented study, we employ two modalities, namely, gene expression and mutational data, to optimize classifiers distinguishing between high and low drug resistance. The study's primary purpose is to showcase that the discrete framework provides a platform for the unification of data, enabling optimization of interpretable multimodal classifiers. In the study, we compare two approaches to the discretization of the IC50 values, namely, using median and quartiles as thresholds. Although the quartile-based approach enables creating a large margin between the

classes, the employed gene expression and mutational data do not allow for satisfactory generalization to novel samples. Here, one could consider extending the maximal size of the classifiers to increase their robustness, as merely ten features may not capture intricate patterns underlying the data. Alternatively, one may adjust other parameters of the algorithm to increase the performance of classifiers. However, the average robustness score ($\Delta_{feature}$) recorded for the gene expression data indicates that most of the features do not separate the classes well. To draw more general conclusions regarding the performance of our approach in terms of drug response prediction, one should consider a more systematic evaluation for multiple different drugs. However, the optimized classifiers are inherently interpretable and may help uncover potentially interesting gene and mutational targets.

In the second exemplary study, we focus on an application beyond the context of cancer research. Here, we employ a data set comprising continuous-valued activation features of a deep neural network trained to distinguish phages following different life cycles. The long-term goal of the study is to identify regions of phage genomes responsible for the particular life cycle. We demonstrate the ability of our approach to reduce the number of features necessary to achieve high prediction accuracy and build an interpretable, human-readable classifier in a short time. In the next step, the selected features can be subject to extensive analysis and interpretation. However, in the case of the identification of life cycle-related genes in phages, incomplete genome annotation pose a major challenge to the recognition of genes of interest. Furthermore, phage genomic sequences can be very similar to each other and cluster into subgroups based on genome similarity, usually following the same life cycle within the cluster. We suspect that creating the training, validation and test data by dividing the genomes on the phage level may cause information leakage. This may result in the network learning, in fact, the sequence similarities instead of functional features (visible also in the particularly good generalization to novel phages). This aspect may be easily addressed by dividing the data by taking the sequence similarity into account, e.g., using phage clusters. Such an approach may pressure the network to learn beyond the sequence similarity and increase the interpretability of classifiers.

Chapter 7

Summary and conclusions

In our work, we exploit the Boolean framework to represent the classifier's architecture and describe the circuit's input-output signal processing. This enables leveraging the advantages of formal methods and utilizing the Answer Set Programming paradigm to design globally optimal solutions in a short time (see Chapter 3). To demonstrate the performance of our approach, we perform simulated and breast cancer data studies and show that our method outperforms the state-of-the-art approach (SynNet) in the binary classification context. In the breast cancer case study, we employ the evaluation metrics described by Mohammadi et al., 2017 and observe a substantial discrepancy between the performance reported using binary and non-binary metrics for classifiers designed with SynNet. In terms of the single-circuit design, the classifiers are represented as Boolean terms that approximate the biochemical model of the circuit. However, the robustness of Boolean approximation decreases if the underlying assumptions are violated. In particular, ambiguous miRNAs that do not separate the classes well and oscillate around the discretization threshold may generate the between-metric discrepancy. In contrast to SynNet, our designs seem to be resistant to uncertain expression patterns and perform with similar accuracy in both binary and non-binary settings. The discrete framework offers a plethora of tools that can support the Boolean-based design of synthetic devices. However, one must account for the existing limitations described further in this chapter to harness the potential of the logic toolkit.

Inspired by ensemble methods, we introduce a novel theoretical design of cell classifiers consisting of multiple circuits (see Chapter 4). The multi-circuit classifiers exploit the advantages of ensemble learning to improve generalization to unseen examples (Opitz and Maclin, 1999b). The primary motivation behind this study was to compare single-circuit and multi-circuit designs. In Chapters 4-5, we show that multi-circuit classifiers outperform single circuits, in particular, facing heterogeneous and novel data. We hypothesize that there are two main reasons behind the better performance of distributed classifiers. First, the multi-circuit classifiers leverage the potential of ensemble learning. Here, the combined predictions of multiple simpler circuits cover a variety of patterns in the data. Further, the threshold function allows for flexibility with regard to the number of circuits that must simultaneously identify the samples belonging to the positive class. In contrast, the single-circuit classifiers specified in the Conjunctive Normal Form require every module to classify a sample as positive to trigger the response. This is reflected in the reported True Positive Rates (see Section 5.4.2) that are substantially lower for most of the cancer data sets in comparison to the multi-circuit designs. Second, although the primary aim of the study was to compare different circuit architectures, the comparison is not isolated from the influence of applying different optimization strategies and computational frameworks. Both approaches substantially differ in the underlying principles as well as the implementation. The GA-based approach employs

active optimization of robust designs in both the parameter tuning procedure (by introducing inner cross-validation) and training (by employing the multi-objective function). We have shown that in the context of cancer expression data, the multi-objective function increases the generalization of the classifiers to novel data.

However, as a heuristic approach, GA does not ensure the optimality of found solutions. Also, the algorithm requires tuning several parameters to perform well in the context of new data. In contrast, the ASP-based approach guarantees that a found solution is an optimal one and does not involve parameter optimization. Also, the ASP enables proving that for a particular data and a set of constraints, a solution does not exist. Although in terms of the generalization error ASP performs worse, potentially due to lack of regularization and thus overfitting the data, one could consider using ASP as a feature extraction method similarly to the studies presented in Chapter 6. We revisit this aspect further in this chapter. In most of the presented cancer case studies, the ASP-based approach returned classifiers in a short time. However, in the case of larger simulated data, the solving run time substantially increases, exceeding the time limits. Thus, for applications involving sizable data, the heuristic approach seems to be more suitable.

Across all the presented studies, we employ discretized data to design cell classifiers. Discretization enables obtaining clear-cut and explainable information about the features in the data and frequently increases the efficiency of the learning process (Gallo et al., 2016a). For instance, in the case of miRNA expression data, the miRNA state is easily interpretable as high (above the discretization threshold) or low (below the discretization threshold). Thus, single-circuit classifiers are inherently explainable and do not require post-hoc processing for human readability. The multi-circuit classifier's decision threshold influences the interpretability of the designs. Besides the threshold, the basic principles of classifier design are shared between the architectures, and thus, the distributed classifiers are also human-readable. Nonetheless, to facilitate the interpretability of the multi-circuit classifiers, we provide rankings of features frequently occurring in the circuits as up- or down-regulated, as well as their robustness scores.

Discretization facilitates the removal of non-relevant features and reduction of noise (Gallo et al., 2016a). Thus, it can also be applied as a feature selection procedure and thus accelerate the training of classifiers. This is displayed across the thesis, as we frequently remove non-relevant features from the data according to the discretization results (Chapters 3-6). Furthermore, discretization enables homogenization of multimodal data, providing a setting for optimizing explainable classifiers comprising features coming from diverse data types. We showcase it in Chapter 6 in a drug response prediction case study, in which we optimize classifiers based on unified gene expression and mutational data.

However, the discretization process is also related to data resolution loss (Gallo et al., 2016a). For instance, in the context of miRNA expression data, we only preserve information about whether a given miRNA is above or under the discretization threshold but lose information about the distance of particular expression values from the threshold. This could be solved by adding a discretization margin similar to the one proposed by Mohammadi et al., 2017. Also, miRNAs differ in their overall discriminative power, namely, the capability of separating samples belonging to different classes. In Chapter 5, we address this aspect by introducing a so-called feature robustness score. The robustness score obtained in the discretization process stores the information about each feature's discriminative power. Thus, we employ it to compute the average robustness of a classifier and embed the score into the

objective function to prioritize features with higher discriminative capability (Chapters 5-6). In Chapter 5, we use expression data to showcase that the multi-objective function enables better generalization of classifiers to novel data.

The estimation of the classifier's performance is as reliable as the employed data, its preprocessing and the evaluation setup. Creating a proper evaluation environment is a non-trivial task. Whalen et al., 2022 describe several parts of study design, where so-called pitfalls commonly lead to incorrect estimation of performance. In Chapters 4 and 5, we address the issues of information leakage, data imbalance and so-called batch effects resulting from different techniques of data acquisition. In particular, we focus on suitable data division into training and validation fractions by minimizing information leakage and, at the same time, preserving the link between the data sets. We also perform a cross-platform data study to investigate how well classifiers trained on a data set generated using one platform generalize to samples obtained with another device. This poses a major challenge, particularly if the sets of features are not covered between the platforms and the data loses its original dimensions.

Except for the above-mentioned aspects of the classifier design, the translation of cell classifier circuits from *in vivo* mouse models (Dastor et al., 2018) to clinical studies also poses many challenges, particularly in the context of personalized therapeutics. Designing the therapeutics for a single patient requires gathering data that would enable optimization of the circuits. However, performing multi-region biopsies from cancerous and non-cancerous tissues is challenging, especially in the case of patients with advanced cancer stage (Dagogo-Jack and Shaw, 2018). Thus, the *in silico* design frameworks that require large amounts of training examples may not be suitable for this particular task. Alternatively, in the future, pre-trained models could be fine-tuned on a reduced set of examples similarly to, for instance, the approaches employed in the protein design domain (Biswas et al., 2021).

Nevertheless, obtaining a nearly complete representation of the tumour's genetic and molecular composition is practically impossible with the currently available techniques. Therefore, the data employed in the classifier's design corresponds to only a partial description of the heterogeneous landscape of cancerous tissue. The circuits delivered to the tumour and the surrounding tissues must be capable of recognizing cells that are incompletely characterized by the data or entirely novel. This poses a major challenge in the circuit's optimization, as the classifiers must generalize to novel samples unseen in training. Thus, in Chapter 5, we focus on creating various scenarios to simulate heterogeneous environments and measure the classifier's performance on hold-out data.

In contrast to single-circuit classifiers, the proposed multi-circuit devices have not been experimentally validated yet. However, logic concepts of similar complexity have been previously explored *in silico* and *in vitro*, e.g., robust platforms implemented with recombinase-based systems or via amplification of signals (Courbet et al., 2015; Chiu and Jiang, 2017). For instance, the decision threshold function can be achieved via amplifying the signal only above a certain threshold using an intermediate compound similar to the synthetic miRNA FF4 introduced by Mohammadi et al., 2017. Also, multi-circuit programs and computing systems with increasing complexity operating in living cells were successfully studied before (Moon et al., 2012; Lapique and Benenson, 2014).

Although we focus on a particular use-case scenario of cell classifiers, the applicability of such circuits goes beyond the cancer context. In particular, similar designs are employed for cell state detection and state-specific targeting of different tissues and cell lineages, e.g., neuronal subtypes or differentiating human pluripotent stem cells (Sayeg et al., 2015; Prochazka et al., 2022). As the cell classifiers utilize inherent and omnipresent biological mechanisms, they allow targeting a plethora of different tissues and cell types, providing a powerful platform for synthetic therapeutic design.

Finally, we present alternative applications of the developed methods to showcase the flexibility of the algorithm proposed in Chapter 5. In the drug response prediction study, we leverage the advantage of the discrete framework to unify multimodal data and optimize classifiers comprising different types of features. Although we do not obtain high prediction accuracy, we propose an interpretable approach for feature extraction. Investing more resources in adjusting the optimization procedures to the needs of the particular application could yield better performance. In the second study comprising the life cycle prediction of bacteriophages, we focus on demonstrating the applicability of the proposed algorithm beyond the cancer context. We achieve nearly identical prediction accuracy as the described deep learning model, simultaneously drastically compressing the classifiers size-wise. The abovementioned examples indicate that the proposed methods may be employed beyond the context of their primary application.

Outlook

In the past decade, the development of synthetic biology accelerated, resulting in the successful creation of many new synthetic devices (Voigt, 2020; Kitano et al., 2023). The rational synthetic design relies on a so-called *design-build-test-learn* cycle to develop novel biological parts and circuits. The cycle starts with an informed, data-driven *design* of the devices. Next, the device must be *built* and *tested* experimentally. By gathering the data and knowledge from the *test* phase, one can *learn* and improve the design. Then, the cycle starts again with redesigning and refining the circuit. Recently, the *design* and *learn* components of the pipeline have been particularly supported by artificial intelligence (Biswas et al., 2021; Hérisson et al., 2022; Yeh et al., 2023). The researchers leverage the increasing accessibility of powerful hardware and software to facilitate the design process, as the assembly and experimental validation are usually time-consuming and expensive to perform (Kitano et al., 2023). A prime example is the rapidly developing field of protein design. For instance, Biswas et al., 2021 follow the development cycle by designing novel proteins based on available data (*design*), followed by their synthesis (*build*) and experimental validation (*test*). The knowledge gathered from the validation is employed to refine the protein sequences (*learn*). In the study, the authors utilize existing databases to train an unsupervised model to design novel sequences and fine-tune it with data obtained from experimental validation. Although the experimental data is scarce, the authors demonstrate that their approach facilitates synthetic protein engineering.

In this thesis, I focus on the *design* component of the mentioned pipeline, developing logic-based machine learning approaches for the design of the synthetic classifier circuits. However, only the experimental validation of the circuit may give the final insight into their performance in a real-world environment. Nevertheless, the

primary motivation behind the development of the proposed Boolean-based frameworks was to select potential candidates for the experimental trials and assess the performance of different classifier topologies. The knowledge gained from the testing process could assist the further development of the computational approaches. However, the cell classifier design problem, or the design of synthetic circuits in general, could also follow an alternative, to some extent contradictory, path. In the presented work, I focus on the rational, data-informed design of synthetic circuits employing Boolean approximations of the circuit's model. As many synthetic part, circuit and device representations, e.g., in the form of sequencing data, are deposited in different databases (e.g., BioParts portal by Plahar et al., 2021), one could consider training an unsupervised deep learning model on the available circuit data to create a general representation of the synthetic circuit designs. Then, such a model could be fine-tuned using (usually scarce) data describing circuits of interest to design the entire circuit without rational design input from the researcher, similarly to the study performed by Biswas et al., 2021. In the future, such an approach could potentially take the place of the laborious trial-and-error validation of the circuit consisting of multiple various components and underlying parameters and advance the further development of the entire field, in particular, if combined with discrete modelling.

In Chapter 6, we consider alternative applications of the proposed classifier design methods. We leverage the inherent interpretability of proposed logic classifiers to combine them with deep learning models as an approach to feature extraction and classifier compression. This direction could be further explored, as in the presented use-case scenario, the compressed classifiers maintain the high accuracy of the trained deep neural network model. As the proposed classifier design approaches rely on the general Boolean logic principles, such methods could be, in fact, applicable to any binary classification problem. The frameworks described in this thesis can also be coupled to leverage the potential of both approaches simultaneously. The ASP-based framework enables finding globally optimal Boolean terms, whereas the GA-based workflow focuses on increasing their robustness. Both approaches can be combined into a two-level strategy comprising (1) pre-optimization of short Boolean terms with the ASP solving protocol by relaxing the constraints in terms of sample misclassification and (2) employing the pre-optimized terms as single classifier components in the generation of the initial population in the genetic algorithm. This procedure, although not yet extensively tested, is implemented and available within both frameworks. The mentioned examples of (re)usage of the logic-based approaches or their components do not exhaust the potential of the discrete environment. In conclusion, the discrete framework provides a versatile platform for designing inherently interpretable classifiers and is applicable to various problems in and beyond computational synthetic biology.

Appendix S2

Chapter 2: Preliminaries

S2.1 Details regarding the biochemical model of classifier circuit

The model employs five biochemical parameters: (1) C_1 being the level of intracellular miRNA resulting in the knockdown of 50% of the target, describing the circuit's response to different levels of input miRNAs, (2) C_2 is the dissociation constant (of the activator tTA from its promoter containing tetracycline response element (pTRE)), (3) T_{max} is the maximal level of the activator rtTA, (4) $FF4_{max}$ is the maximal level of synthetic miRNA miR-FF4, and (5) Out_{max} - the maximal level of the output. Table S2.1 includes the parameter values estimated for particular binarization thresholds. Two of the above-mentioned parameters (C_2, T_{max}) are specific for the binarization threshold used to discretize continuous miRNA levels in a given data set. These parameters can be adjusted in the lab to the chosen binarization threshold. The rest of the parameters (C_1, $FF4_{max}$, Out_{max}) are constant regardless of the chosen binarization threshold.

THRESHOLD	C1	C2	Tmax	FF4max	Outmax
θ_1 (50 mol/cell)	20	18557	5389	3000	50000
θ_2 (250 mol/cell)	20	10251	9755	3000	50000
θ_3 (1250 mol/cell)	20	5340	18727	3000	50000

TABLE S2.1: Values for the parameters for different binarization thresholds.

Appendix S3

Chapter 3: Discrete Cell Classifiers

S3.1 Removal of isomorphic solutions in a post-processing step

Within a set of optimal solutions returned by the ASP solver, we first sort each classifier's gates by the inputs' IDs. Then we rewrite all the solutions by assigning to each gate an ID in ascending order preserving the original gate-to-input relation. The input and gate IDs are then ordered identically for all the isomorphic solutions in each class, which makes them indistinguishable. As isomorphic solutions perform with identical prediction accuracy, an arbitrary representative can be chosen as the final solution.

Data set	θ	Circuit
All	θ_2	$\neg miR\text{-}144 \wedge \neg miR\text{-}320\text{-}RNASEN \wedge \neg miR\text{-}99a \wedge miR\text{-}21$
Triple-	θ_2	$\neg miR\text{-}376c \wedge \neg miR\text{-}378 \wedge \neg miR\text{-}451\text{-}DICER1$
Her2+	θ_3	$\neg miR\text{-}376a\text{-}1\text{-}3p \wedge \neg miR\text{-}376c \wedge \neg miR\text{-}144 \wedge (miR\text{-}21 \vee miR\text{-}375) \wedge (miR\text{-}21 \vee miR\text{-}7\text{-}1)$
ER+ Her-	θ_3	$\neg miR\text{-}483\text{-}1\text{-}5p \wedge \neg miR\text{-}296\text{-}3p \wedge \neg miR\text{-}574\text{-}5p \wedge (miR\text{-}21 \vee miR\text{-}105\text{-}1) \wedge (miR\text{-}21 \vee miR\text{-}1251)$
Cell Line	θ_1	$\neg miR\text{-}376c \wedge \neg miR\text{-}145 \wedge \neg miR\text{-}451\text{-}DICER1$

TABLE S3.1: Pruned circuits presented in Mohammadi et al., 2017 for five breast cancer data sets (adapted Table S5 in Mohammadi et al., 2017). θ 1-3 states for different sets of biochemical parameters used for optimization related to the applied binarization threshold presented in Mohammadi et al., 2017.

S3.2 Example 1: A thin line between the groups and false predictions

Sample	Annot	Output [mol/cell]
2	0	176,34
4	0	197,14
9	0	230,17
11	0	242,80
5	0	299,50
8	0	305,70
3	0	500,16
6	0	539,70
1	0	560,77
10	0	625,07
7	0	660,09
178	1	662,82
49	1	678,48
169	1	696,05
141	1	697,54
19	1	724,74
36	1	735,50
146	1	743,16
35	1	745,35
21	1	787,90

TABLE S3.2: The circuit output concentrations for the first 20 samples (11 negative and 9 positive) sorted by the circuit output concentration for the Breast Cancer All data set. The groups are separated with a bold line.

S3.3 Example 2: Misclassification of positive samples in Triple- data set

In Table S3.3, we present real-valued concentrations of miRNAs occurring in the classifier. The binarization threshold applied for the Triple- data set is 250 copies/-cell. For example, we consider sample 73 annotated as positive to compute the output concentration. All the miRNAs occurring in the Boolean classifier are expected to be low expressed (below the threshold) in a positive sample to classify that sample as diseased. However, according to the concentration of miR-378 and the binarization threshold, the miRNA in the sample is upregulated. In this case, the Boolean classifier output is equal to 0 ($\neg 0 \wedge \neg 1 \wedge \neg 0 = 0$) and the sample is misclassified as a false negative. The only scenario for the function to output 1 (positive/cancerous) is if all the miRNAs are downregulated (0). Respecting the model and the repressing function of the NOT gate, if miRNA-378 is upregulated according to the threshold, it should target the corresponding site and block the production of the output protein, which should result in the survival of the diseased cell.

Here, we calculate the circuit output concentration for the presented classifier for sample 73 to validate the output value estimation. We assume that $FF4 = 0$ as the

Sample	Annotation	Output [mol/cell]
6	0	58,31
9	0	425,81
5	0	449,05
2	0	491,84
11	0	537,70
4	0	579,37
7	0	999,29
10	0	1026,33
3	0	1188,47
1	0	1367,58
8	0	1379,61
73	1	1671,57
71	1	2068,85
56	1	2169,57
30	1	2340,93
82	1	2400,44

TABLE S3.3: The circuit output concentrations for the first 16 samples (11 negative and 5 positive) sorted by the circuit output concentration for the Breast Cancer Triple- data set. The groups are separated with a bold line. Exemplary samples misclassified by the Boolean classifier are highlighted in light red.

Sample	*miR*-376*c*			*miR*-378			*miR*-451-*DICER*1		
	C	B_e	B_o	C	B_e	B_o	C	B_e	B_o
73	80.67	0	0	317.56	0	1	180.01	0	0
71	4.72	0	0	30.48	0	0	428.16	0	1
56	8.32	0	0	284.31	0	1	148.29	0	0
82	10.48	0	0	96.46	0	0	289.65	0	1

TABLE S3.4: Breast Cancer Triple-: evaluation of miRNAs continuous levels (C), desired boolean value (B_e) and the output boolean value according to the threshold (B_o).

circuit does not include OR gates. According the model presented in Mohammadi et al., 2017 the value of f_1 is calculated as follows:

$$\begin{aligned} f_1(miR\text{-}376c, miR\text{-}378, miR\text{-}451\text{-}DICER1) = \\ 1 - \frac{0 + 80.67 + 317.56 + 180.01}{0 + 20 + 80.67 + 317.56 + 180.01} \\ \approx 0.033431398 \end{aligned} \tag{S3.1}$$

For this particular circuit, only two biochemical parameters are relevant: C_1 and Out_{max}, which are constant. This implies that one could potentially apply any binarization threshold for the data set as it does not influence the output concentration. The $C(\theta_2, R_{Triple-}, g_{Triple-})$ is calculated as follows:

$$C(\theta_2, R_{Triple-}, g_{Triple-}) = 50000 \cdot 0.033431398 \approx 1671.57 \tag{S3.2}$$

Even though the samples are perfectly separated (AUC = 1.0), the Boolean representation does not agree with the annotation of the samples based on the analysis of the binarized data set. The specification of the model does not reveal whether some of the miRNAs forming the classifier are upregulated according to the given threshold while expected to be downregulated. In fact, the f_1 function summarizes the continuous concentrations of miRNAs but does not reflect the applied threshold.

S3.4 Example 3: Misclassification of negative samples Her2+ data set

Table S3.5 contains output values for three misclassified negative samples (3 in total for the 'emphHer2+ data set). As the sample is classified as negative (healthy), the output production must be blocked to protect healthy cells from apoptosis. Based on the f_1 function (see Mohammadi et al., 2017), one expects that at least one of the miRNAs forming the NOT gates is upregulated and blocks the production of the toxic output compound directly. Otherwise, at least in one OR gate, both miRNA inputs should be downregulated to activate miR-FF4. According to the binarization threshold, all the miRNAs forming NOT gates are observed to be downregulated (Table S3.7). Furthermore, in both OR gates, *miR*-21 is upregulated according to the threshold (Table S3.6). As an example, we consider sample 6 and calculate the circuit output concentration. The output of the boolean function for sample 6 according to the threshold is 1 $((1 \vee 0) \wedge (1 \vee 0) \wedge \neg 0 \wedge \neg 0 \wedge \neg 0 = 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 = 1)$ and does not agree with the annotation.

Here we validate the calculation of the output concentration for sample 6. First, the concentration of miR-FF4 is calculated in two steps. The f_1 is calculated as follows:

$$\begin{aligned} &f_1(miR\text{-}21, miR\text{-}375, miR\text{-}21, miR\text{-}7\text{-}1) = \\ &1 - \frac{1454.46 + 1.47 + 1454.46 + 2.54}{20 + 1454.46 + 1.47 + 1454.46 + 2.54} \\ &\approx 0.02709173956 \end{aligned} \tag{S3.3}$$

The f_2 is calculated with $C_2 = 5340$ and $T_{max} = 18727$ as in Mohammadi et al., 2017:

$$f_2(0.02709173956) = \frac{0.02709173956}{\frac{5340}{18727} + 0.02709173956} \approx 0.08676533241 \tag{S3.4}$$

Finally, $FF4$ is calculated as follows:

$$FF4 = 3000 \cdot 0.08676533241 \approx 260.2959972 \tag{S3.5}$$

The influence of the NOT gates is calculated as in the previous example:

$$\begin{aligned} &f_1(FF4, miR\text{-}376a\text{-}1\text{-}3p, miR\text{-}376c, miR\text{-}144) = \\ &1 - \frac{260.2959972 + 20.84 + 20.84 + 765.22}{20 + 260.2959972 + 20.84 + 20.84 + 765.22} \\ &\approx 0.01839594705 \end{aligned} \tag{S3.6}$$

Sample	Annotation	Output concentration [mol/cell]
11	0	640,34
5	0	871,50
4	0	903,07
9	0	913,71
6	0	919,80
7	0	1100,02
8	0	1264,97
1	0	1364,83
2	0	1847,34
10	0	2271,71
3	0	2298,18
86	1	3467,77
30	1	3882,19
67	1	4062,07
23	1	4187,43
53	1	4453,42
32	1	4733,44
57	1	4944,93
84	1	5090,99
12	1	5448,82

TABLE S3.5: The circuit output concentrations for the first 20 samples (11 negative and 9 positive) sorted by the circuit output concentration for the Her2+ data set. The groups are separated with a bold line.

Sample	*miR*-21			*miR*-375			*miR*-7-1		
	C	B_e	B_o	C	B_e	B_o	C	B_e	B_o
6	1454.46	0	1	1.47	0	0	2.54	0	0
10	1258.03	0	1	8.23	0	0	1.83	0	0
3	3163.71	0	0	15.55	0	0	0.10	0	0

TABLE S3.6: Her2+, miRNAs in OR gates: Evaluation of miRNAs continuous levels (C), expected Boolean value (B_e) and the actual Boolean value according to the threshold (B_o).

Sample	*miR*-376*a*-1-3*p*			*miR*-376*c*			*miR*-144		
	C	B_e	B_o	C	B_e	B_o	C	B_e	B_o
6	20.84	1	0	20.84	1	0	15.55	1	0
10	39.79	1	0	64.28	1	0	20.46	1	0
3	60.21	1	0	213.06	1	0	15.55	1	0

TABLE S3.7: Her2+, miRNAs in NOT gates: evaluation of miRNAs continuous levels (C), expected Boolean value (B_e) and the actual Boolean value according to the threshold (B_o).

$$C(\theta_3, R_{Her2+}, g_{Her2+}) = 50000 \cdot 0.01839594705 \approx 919.7973526 \tag{S3.7}$$

Again, the model confirms the perfect separation of samples, while according to the Boolean representation and the annotation, a few samples are misclassified.

Appendix S4

Chapter 4: From single- to multi-circuit classifiers

Algorithm S4.1 The algorithm describes random generation of an initial population of p_{size} individuals. The number of rules in each individual is randomly chosen and the rules are randomly generated. After specifying the size of an individual (line 2), c rules must be generated in a few steps. First, the size of a rule (r_{size}) and miRNA IDs ($miRNAs$) must be randomly chosen (lines 4-5). Then, the sign (positive/negative) is randomly assigned to the miRNAs (line 6). Note that in case of r_{size}=2, the miRNAs are connected with an AND gate. c rules generated as described above create an individual which may be added to a population (line 10). The steps are repeated until the population consists of p_{size} individuals (lines 1-11).

Algorithm 1: Initialization of a first population.

```
   Data: dataset D
   Parameters: population size p_size, maximal size of a DC c_max
   Output: Population
 1 for i = 1 to p_size do
       /* randomly choose the size of a new classifier */
 2     c ⟵ RandomlyChooseInRange(1, c_max);
 3     for i = 1 to c do
           /* randomly choose the size of a new rule */
 4         r_size ⟵ RandomlyChooseInRange(1, 2);
           /* randomly choose miRNA IDs */
 5         miRNAs ⟵ RandomlyChooseIDs(D, r_size);
           /* randomly assign miRNA signs */
 6         miRNAs ⟵ RandomlyAssignSigns();
           /* create a new rule */
 7         Rule ⟵ CreateARule(miRNAs);
           /* add a new rule to a classifier */
 8         Individual ⟵ Add(Rule);
 9     end
       /* add a new classifier to a population */
10     Population ⟵ Add(Individual);
11 end
```

Algorithm S4.2 The algorithm describes the selection of parents that are potential candidates to recombine. The parents are chosen in tournaments of size t_{size}, i.e., t_{size} candidates are randomly chosen from the population to participate in a tournament

(lines 1-5, 7-11). In each round 2 parents are selected from the population. The winning candidates are individuals with the highest BACC (lines 5,11). After the first parent is selected its ID is temporarily blocked to be re-selected (line 6). This allows keeping diversity in the population. The new population of selected parents is then utilized to perform crossover.

Algorithm 2: Selection of parents

Input: *Population*
Parameters: population size p_{size}, tournament size t_{size}
Output: $Parent_1$, $Parent_2$

```
   /* repeat adding to a tournament t_size times */
1  for i = 1 to t_size do
       /* randomly choose an individual's ID */
2      Candidate ⟵ RandomlyChooseInRange(1, p_size);
       /* add a candidate ID to a tournament */
3      Candidates ⟵ Add(Candidate);
4  end
   /* choose the best parent in a tournament */
5  Parent_1 ⟵ SelectBest(Candidates);
   /* block a chosen ID to be re-selected */
6  p_size ⟵ BlockID(Parent_1, p_size);
7  for i = 0 to t_size do
       /* randomly choose an individual's ID */
8      Candidate ⟵ RandomlyChooseInRange(1, p_size);
       /* add a candidate ID to a tournament */
9      Candidates ⟵ Add(Candidate);
10 end
   /* choose the best parent in a tournament */
11 Parent_2 ⟵ SelectBest(Candidates);
```

Algorithm S4.3 The algorithm describes the crossover procedure performed on the population of selected parents. Each two parents chosen randomly from the population of selected parents exchange genes with the probability c_{prob}. If the randomly chosen p is lower than c_{prob} the parents undergo the crossover (lines 2-13). Otherwise, the parents are copied directly to a new population (line 15). If parents are of the same size, uniform crossover is performed (line 11-12). Otherwise, index-based crossover is applied (lines 7-9). Both procedures are described in details in section 4.2.6.

Algorithm S4.4 The algorithm describes the index-based crossover that we apply if the sizes of parents differ to preserve a chance for each rule to be exchanged. Here, the rules from the first and second parent are paired off according to a randomly chosen index specifying the position of a shorter parent in relation to the other one. The index is in range between 1 and $ParentSize_1$-$ParentSize_2$ (line 7). Paired rules are crossovered uniformly. Rules that cannot be paired (due to different sizes) may be copied to a randomly chosen child. As a result, the number of rules in each child is between the minimum and the maximum size of the two parents. Note, the index-based crossover may shorten the size of an individual as additional rules cannot be copied to the larger classifier.

Algorithm 3: Crossover

Input: $Parent_1$, $Parent_2$, crossover probability c_{prob}
Output: $Child_1$, $Child_2$
`/* randomly choose the probability of crossover */`
$p \longleftarrow$ DrawProbability(0,1);
`/* if` $p \leq c_{prob}$ `perform crossover */`
if $p \leq c_{prob}$ **then**
  `/* assign a longer parent to` $Parent_1$ `*/`
  $Parent_1$, $Parent_2 \longleftarrow$ AssignParentsBySize($Parent_1$, $Parent_2$);
  `/* assign sizes of parents */`
  $ParentSize_1 \longleftarrow$ Size($Parent_1$);
  $ParentSize_2 \longleftarrow$ Size($Parent_2$);
  `/* if parents sizes differ perform index-based crossover */`
  if $ParentSize_1 \neq ParentSize_2$ **then**
    `/* randomly choose the crossover index */`
    $CrossoverIndex \longleftarrow$ RandomlyChooseInRange(1, $ParentSize_1$ - $ParentSize_2$);
    `/* perform index based crossover */`
    $Child_1$, $Child_2 \longleftarrow$ IndexCrossover($Parent_1$, $Parent_2$, $ParentSize_1$, $ParentSize_2$, $CrossoverIndex$);
    $Population \longleftarrow$ Add($Child_1$, $Child_2$)
  else
    `/* if parents have identical size perform uniform crossover */`
    $Child_1$, $Child_2 \longleftarrow$ UniformCrossover($Parent_1$, $Parent_2$);
    `/* add children to a new population */`
    $Population \longleftarrow$ Add($Child_1$, $Child_2$)
  end
else
  `/* if` *probability* $> c_{prob}$ `copy parents to a new population */`
  $Population \longleftarrow$ Add($Parent_1$, $Parent_2$)
end

Algorithm S4.5 The algorithm describes mutation. Mutation may occur on two levels: both, rules and inputs may mutate. A rule may (i) be removed from a classifier, (ii) be added to a classifier and (iii) be copied from one classifier to another (lines 5-17). An input may (i) be removed from a rule, (ii) be added to a rule, (iii) may change the sign i.e., become a negative or positive input (lines 19-32). Rules, being larger components affecting the classifier size, mutate with a lower probability than inputs (0.2). Note, the maximal size of a classifier (c_{max}) must be preserved.

Algorithm 4: Index-based crossover

Input: $Parent_1$, $Parent_2$, $ParentSize_1$, $ParentSize_2$, $CrossoverIndex$
Output: $Child_1$, $Child_2$

1 **for** $i = 1$ **to** $ParentSize_1$ **do**
  /* decide whether the rule will be exchanged */
  /* $SwapMask$=1 corresponds to rule exchange */
  /* $SwapMask$=0 corresponds to copying without exchanging */
2 $SwapMask \longleftarrow$ RandomlyChooseInRange(0, 1);
  /* 1 - rule is exchanged */
3 **if** $SwapMask = 1$ **then**
    /* if the rules do not pair off */
    /* i.e., there is no possibility to exchange rules */
4 **if** $i < CrossoverIndex$ *OR* $i \geq CrossoverIndex + ParentSize_2$ **then**
      /* copy a rule from $Parent_1$ to $Child_2$ */
5 $Child_2 \longleftarrow$ CopyRule($Parent_1$, i);
6 **else**
      /* if the rules pair off exchange rules */
      /* copy a rule from $Parent_2$ to $Child_1$ */
7 $Child_1 \longleftarrow$ CopyRule($Parent_2$, i);
      /* copy a rule from $Parent_1$ to $Child_2$ */
8 $Child_2 \longleftarrow$ CopyRule($Parent_1$, i);
9 **end**
10 **else**
    /* 0 - rule is not exchanged */
11 **if** $i < CrossoverIndex$ *OR* $i \geq CrossoverIndex + ParentSize_2$ **then**
      /* copy a rule from $Parent_1$ to $Child_1$ */
12 $Child_1 \longleftarrow$ CopyRule($Parent_1$, i);
13 **else**
      /* else copy rules to the parents without exchanging */
      /* copy a rule from $Parent_1$ to $Child_1$ */
14 $Child_1 \longleftarrow$ CopyRule($Parent_1$, i);
      /* copy a rule from $Parent_2$ to $Child_2$ */
15 $Child_2 \longleftarrow$ CopyRule($Parent_2$, i);
16 **end**
17 **end**
18 **end**

Algorithm 5: Mutation

Input: *Population*, maximal size of a DC c_{max}
Output: *Population*

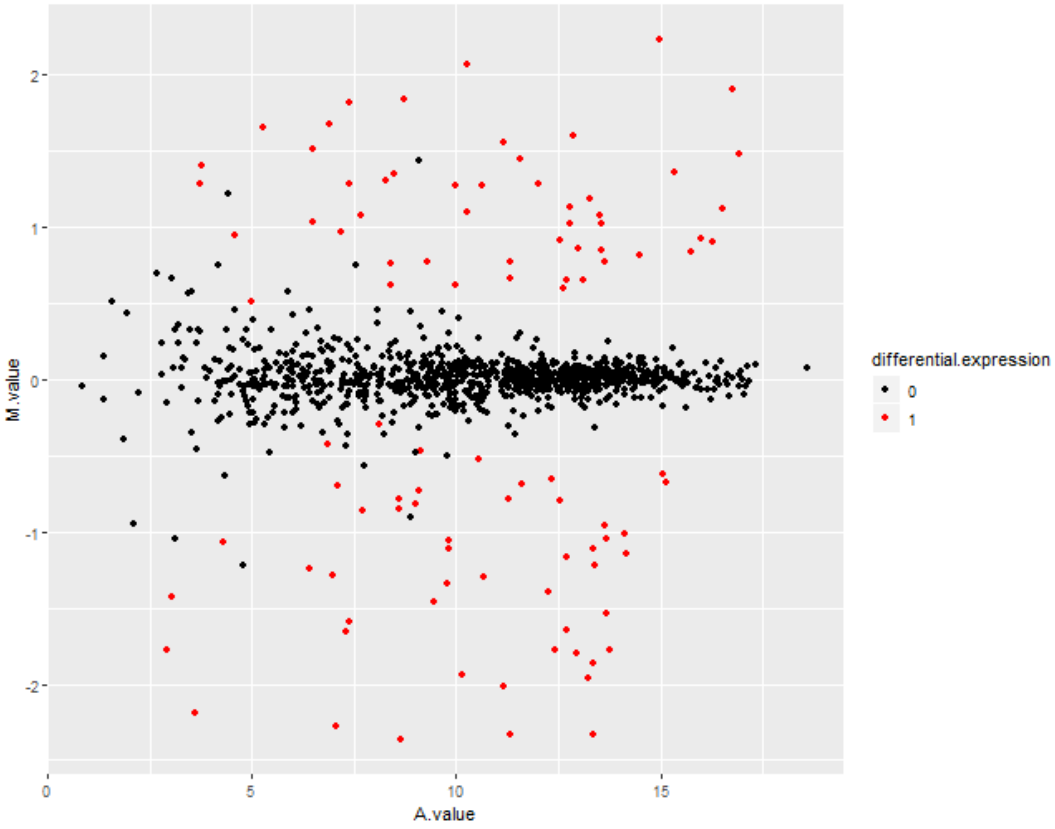
for $i = 1$ **to** p_{size} **do**
  `/* randomly choose the probability of mutation */`
  probability ⟵ DrawProbability(0,1);
  `/* if` *probability* $\leq m_{prob}$ `perform mutation */`
  if *probability* $\leq m_{prob}$ **then**
    `/* choose the mutation level */`
    `/* 1 corresponds to mutation of a rule */`
    `/* 2-4 corresponds to mutation of an input */`
    MutationLevel ⟵ RandomlyChooseInRange(1, 5);
    if *MutationLevel* = *1* **then**
      `/* choose the mutation type */`
      MutationType ⟵ DrawItem(*add*, *remove*, *copy*);
      switch *MutationType* **do**
        case *add* **do**
          `/* add rule */`
          AddRule(*Population*, *i*, c_{max})
        end
        case *copy* **do**
          `/* copy rule */`
          CopyRule(*Population*, *i*, c_{max})
        end
        case *remove* **do**
          `/* remove rule */`
          RemoveRule(*Population*, *i*)
        end
      end
    else
      `/* choose the mutation type */`
      MutationType ⟵ DrawItem(*add*, *remove*, *sign*);
      switch *MutationType* **do**
        case *add* **do**
          `/* add input */`
          Rule ⟵ DrawRule(1, *ps*);
          AddInput(*Population*, *i*, *Rule*)
        end
        case *remove* **do**
          `/* remove input */`
          RemoveInput(*Population*, *i*, c_{max}, *Rule*)
        end
        case *sign* **do**
          `/* change sign of an input */`
          ChangeInputSign(*Population*, *i*, *Rule*)
        end
      end
    end
  end
end

TABLE S4.1: Results of 3-fold cross-validation. Metrics: TPR - average True Positive Rate recorded for validation data in the 3-fold CV, TNR - True Negative Rate, ACC - accuracy, BACC - balanced accuracy, $BACC_{tr}$ - average balanced accuracy recorded for training data in the 3-fold CV.

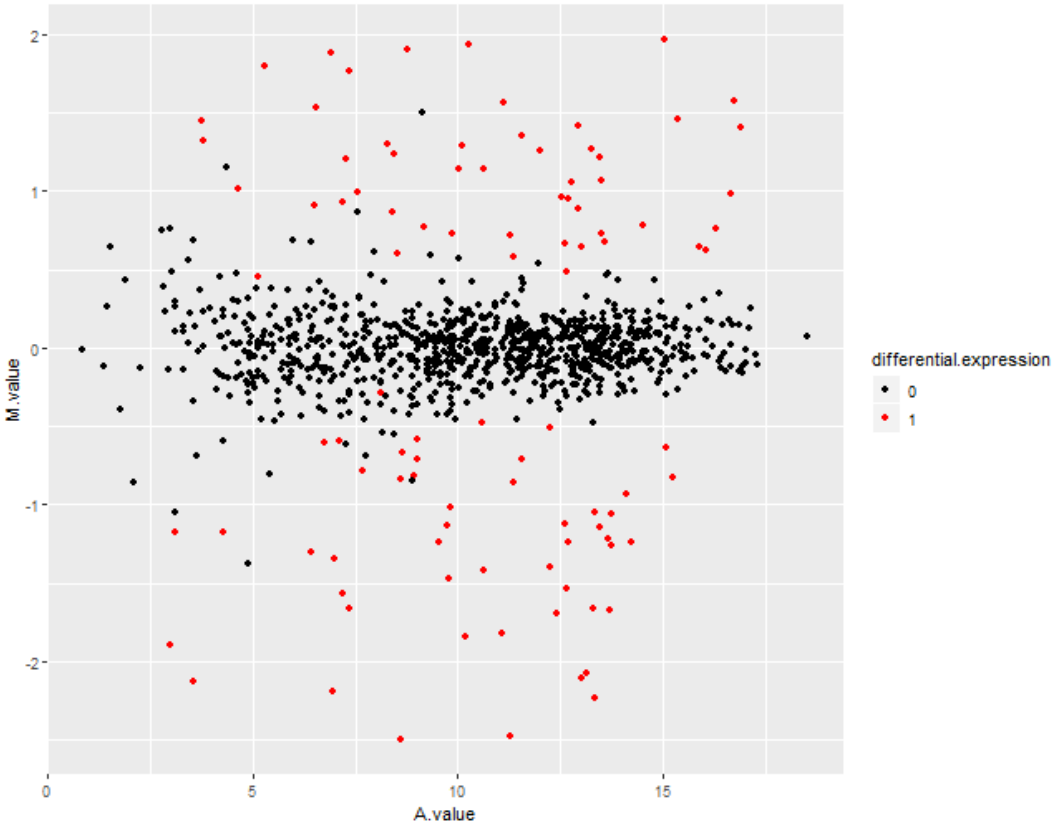
Dataset	α	TPR	TNR	ACC	BACC	$BACC_{tr}$
All	0.85	0.89	0.83	0.88	0.86	0.96
	0.75	0.94	0.81	0.93	0.87	0.98
	0.65	0.95	0.72	0.93	0.83	0.98
	0.60	0.92	0.92	0.92	**0.92**	0.98
	0.50	0.92	0.92	0.92	**0.92**	0.98
	0.40	0.94	0.64	0.92	0.79	0.99
	0.35	0.97	0.72	0.96	0.85	0.99
	0.25	0.96	0.72	0.94	0.84	1.00
Triple-	0.85	0.92	0.75	0.89	**0.83**	0.98
	0.75	0.96	0.67	0.92	0.81	0.99
	0.65	0.94	0.58	0.89	0.76	1.00
	0.60	0.93	0.64	0.89	0.78	1.00
	0.50	0.93	0.64	0.89	0.78	1.00
	0.40	0.94	0.56	0.89	0.75	1.00
	0.35	0.94	0.53	0.89	0.74	0.99
	0.25	0.94	0.53	0.89	0.74	1.00
Her2+	0.85	0.99	0.44	0.92	0.72	0.96
	0.75	0.99	0.61	0.94	**0.80**	0.96
	0.65	0.99	0.53	0.93	0.76	0.96
	0.60	1.00	0.53	0.94	0.76	0.96
	0.50	1.00	0.53	0.94	0.76	0.96
	0.40	0.99	0.53	0.93	0.76	0.96
	0.35	1.00	0.53	0.94	0.76	0.96
	0.25	1.00	0.53	0.94	0.76	0.93
ER+ Her-	0.85	0.90	0.64	0.82	**0.77**	0.93
	0.75	0.90	0.64	0.82	**0.77**	0.93
	0.65	0.90	0.64	0.82	**0.77**	0.93
	0.60	0.90	0.64	0.82	**0.77**	0.93
	0.50	0.90	0.64	0.82	**0.77**	0.93
	0.40	0.90	0.53	0.78	0.72	0.93
	0.35	0.90	0.64	0.82	**0.77**	0.93
	0.25	0.90	0.53	0.78	0.72	0.91
Cell Line	0.85	0.67	1.00	0.87	0.83	1.00
	0.75	0.67	1.00	0.87	0.83	1.00
	0.65	0.67	1.00	0.87	0.83	1.00
	0.60	0.67	1.00	0.87	0.83	1.00
	0.50	0.67	1.00	0.87	0.83	1.00
	0.40	0.67	1.00	0.87	0.83	1.00
	0.35	1.00	0.89	0.93	0.94	1.00
	0.25	1.00	1.00	1.00	**1.00**	1.00

Appendix S5

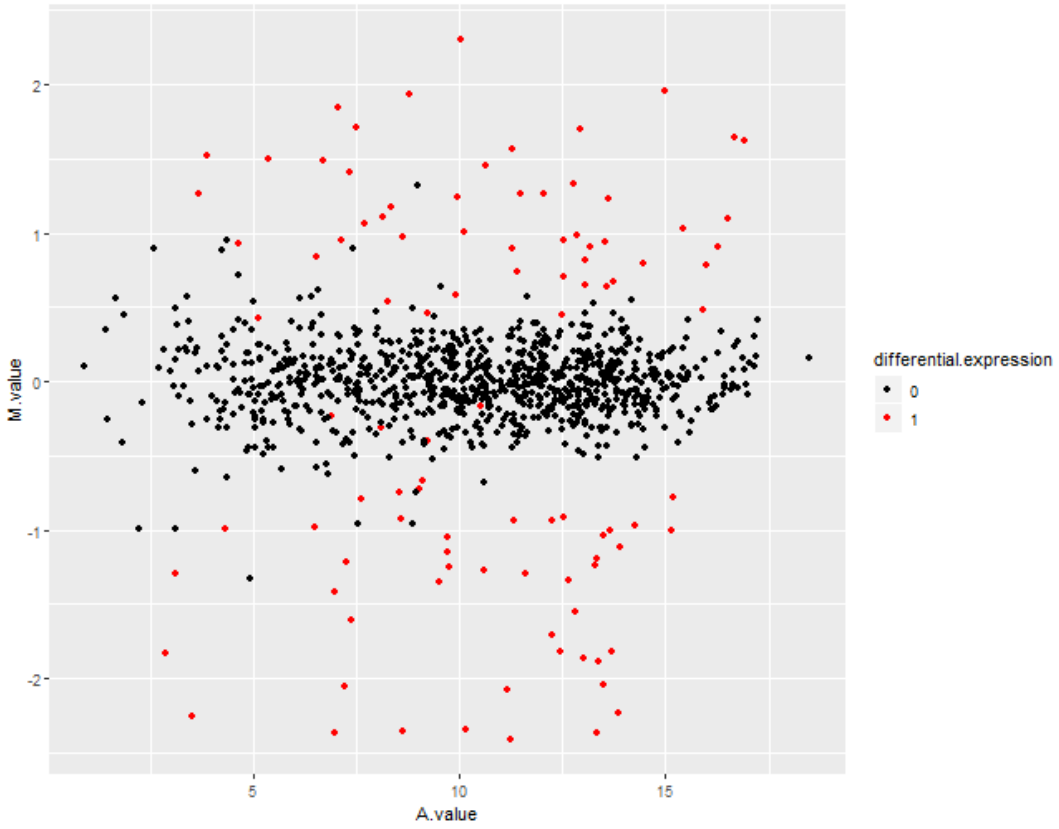
Chapter 5: Finding robust classifiers



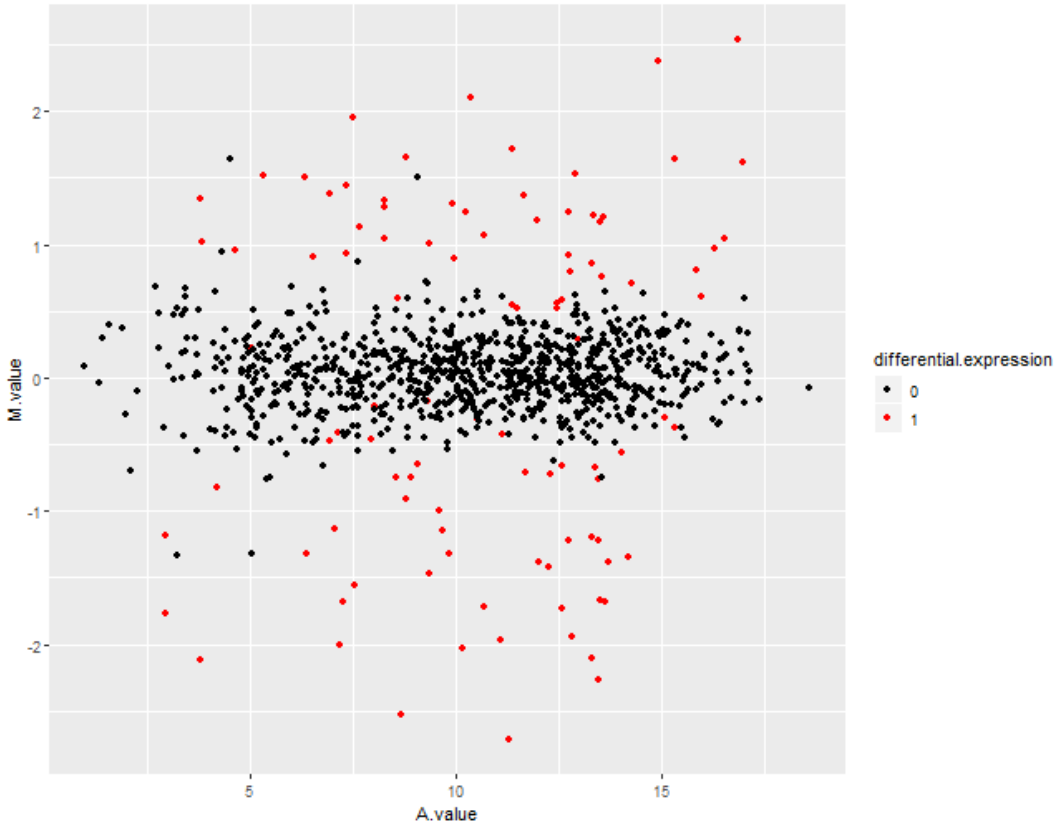
(A) SDR0



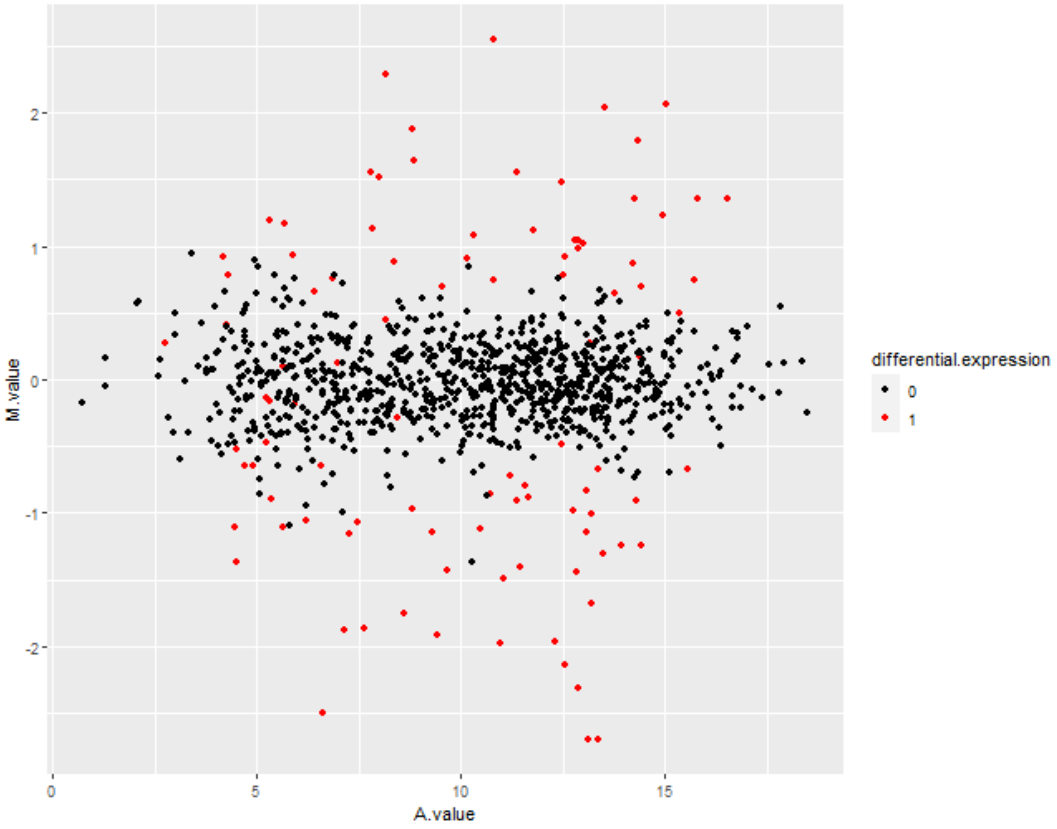
(B) SDR10



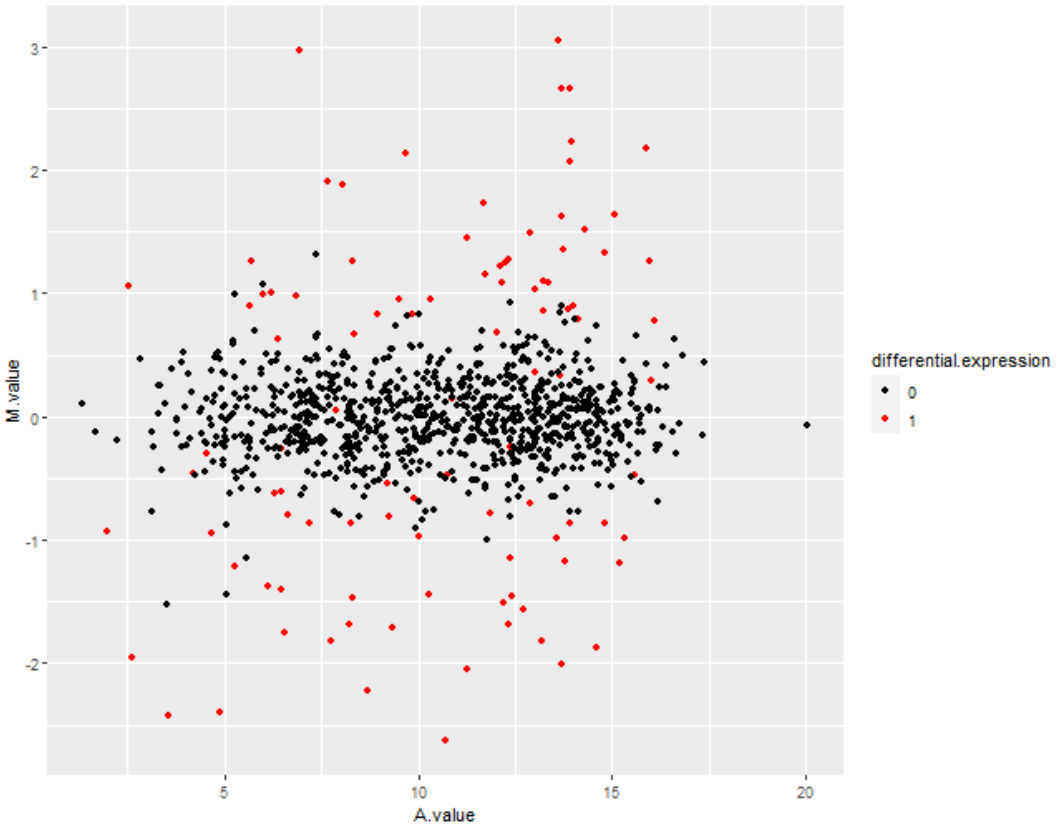
(C) SDR20



(D) SDR30

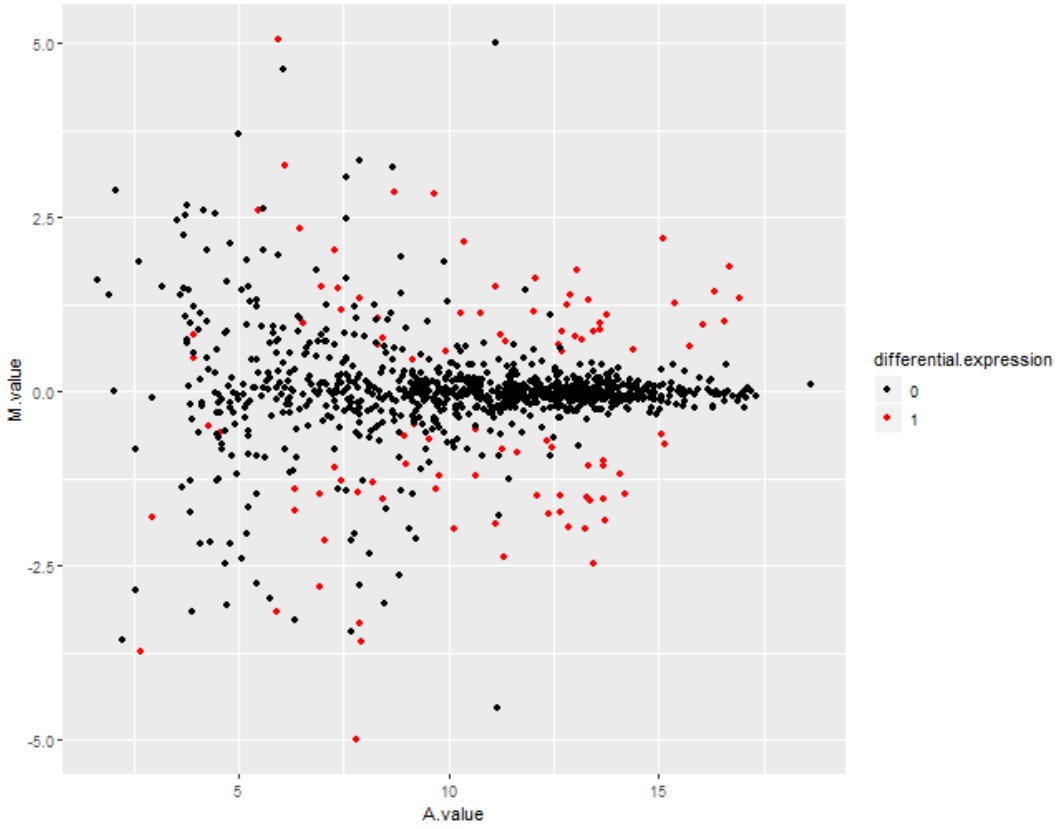


(E) SDR40

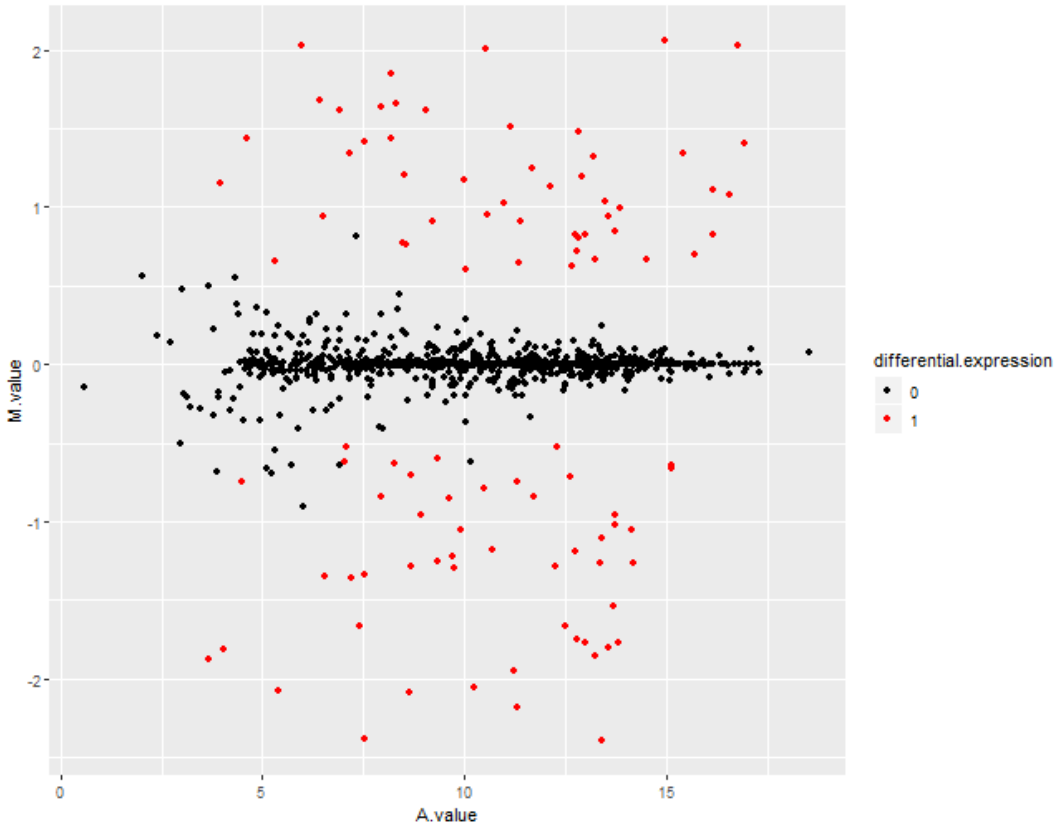


(F) SDR50

FIGURE S5.1: MA plots for the outlier simulated data sets: SDR0, SDR10, SDR20, SDR30, SDR40 and SDR50. Red dots represent truly differentially expressed features and black dots - features that are not differentially regulated. Plots generated with the compCode R package (Soneson, 2014).



(A) SDDISP



(B) SDP50

FIGURE S5.2: MA plots for the SDDISP and SDP50 data sets. Red dots represent truly differentially expressed features and black dots - features that are not differentially regulated. Plots generated with the compcodeR package (Soneson, 2014).

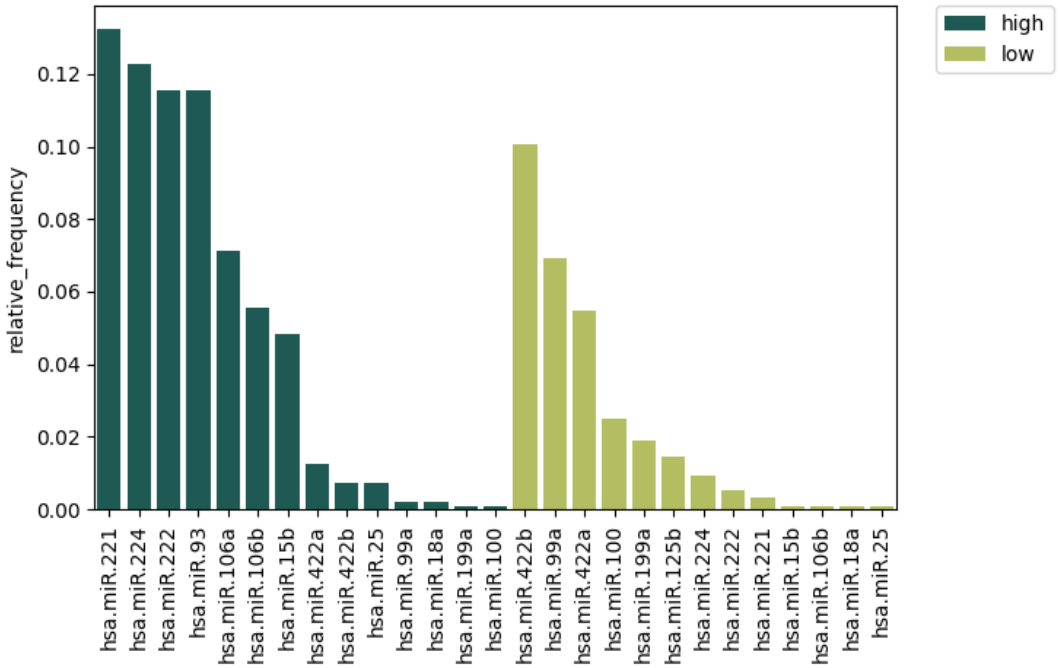
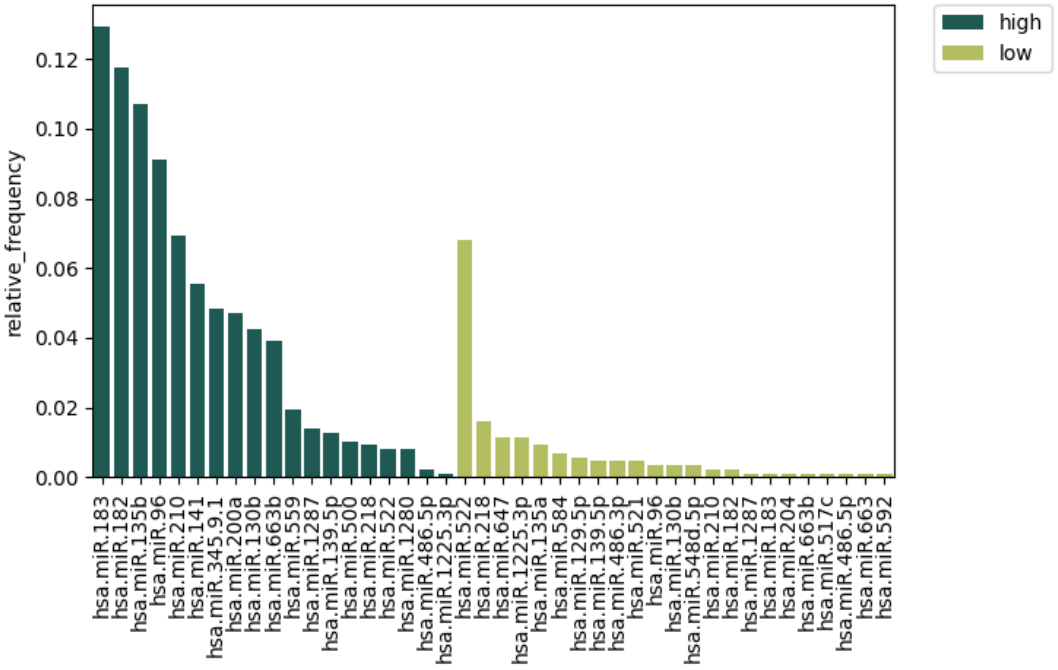
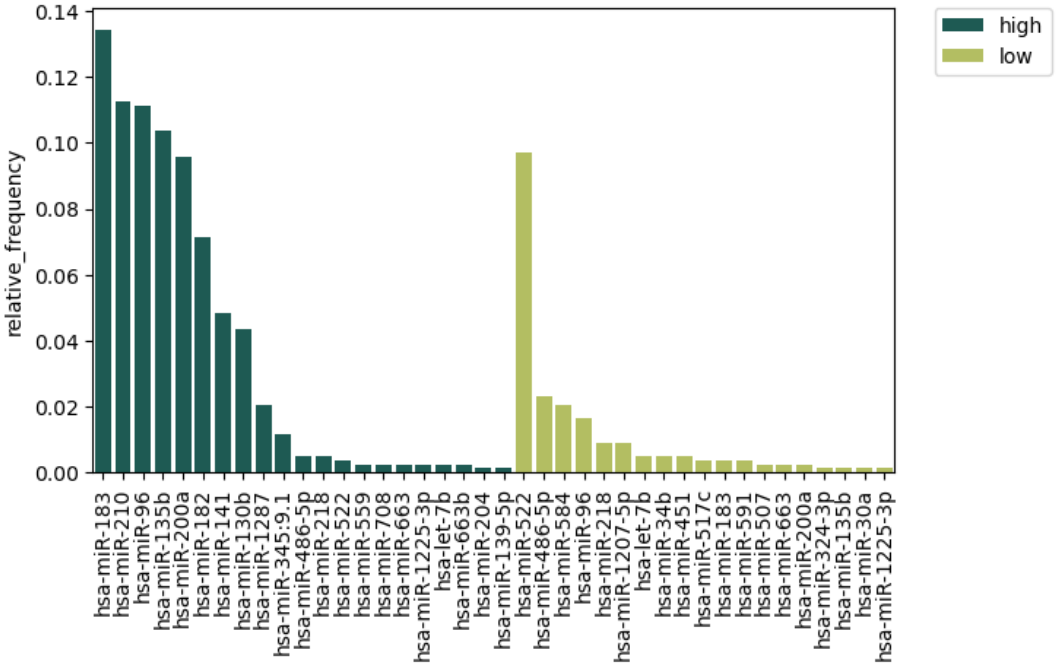


FIGURE S5.3: Frequencies of features in classifiers recorded for jointly normalized GSE10694 data set.

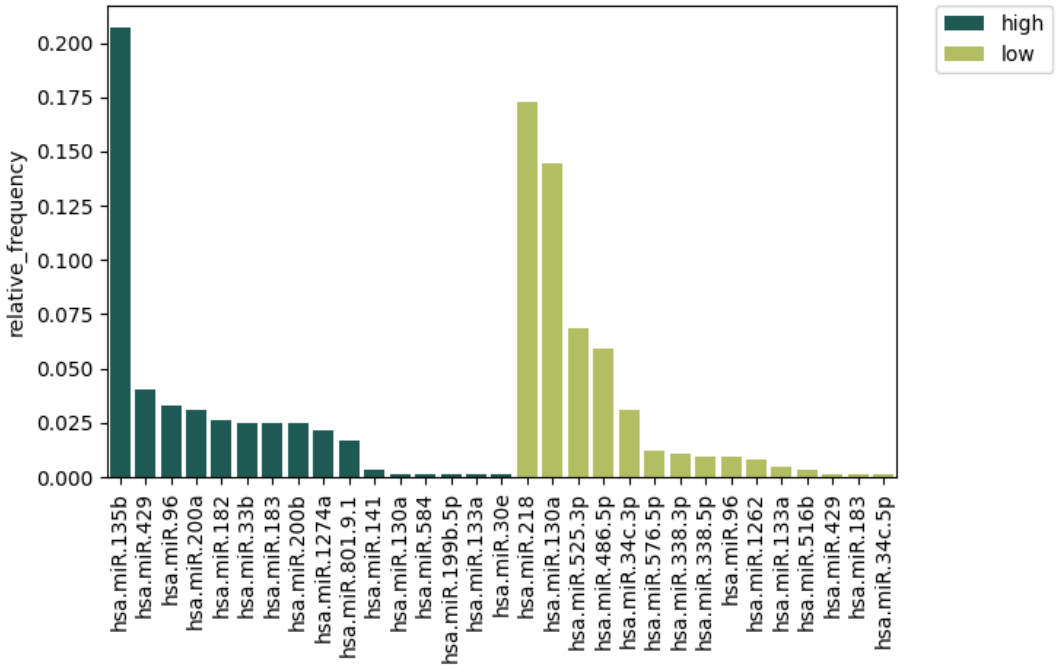


(A) Jointly normalized GSE36681-FF data set.

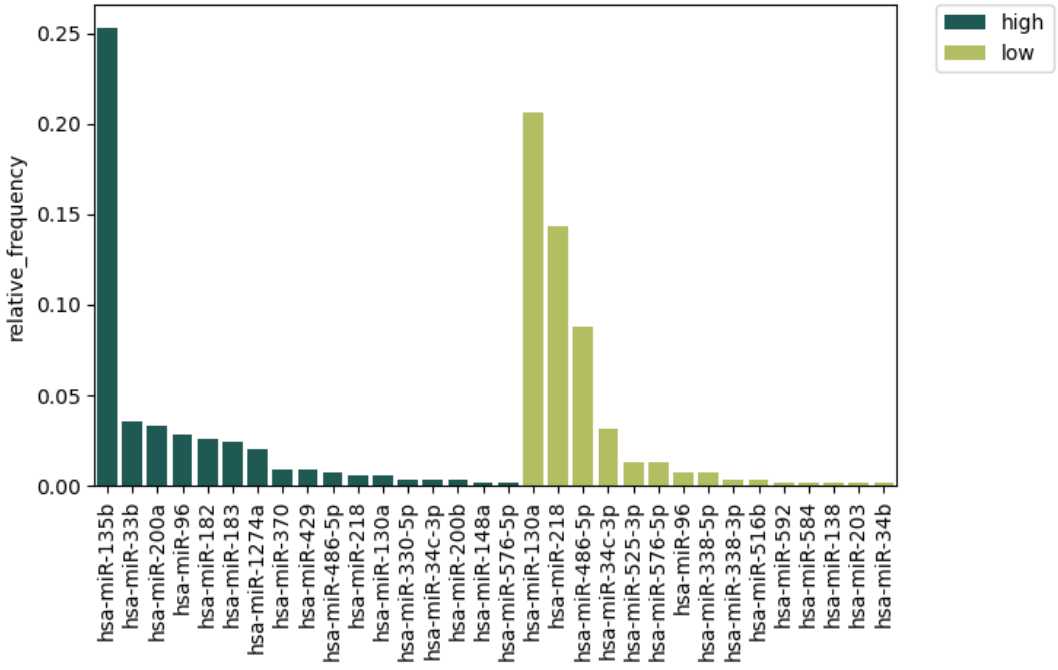


(B) Separately normalized GSE36681-FF data set.

FIGURE S5.4: Frequencies of features in classifiers recorded for (A) jointly and (B) separately normalized GSE36681-FF data set.



(A) Jointly normalized GSE36681-FFPE data set.



(B) Separately normalized GSE36681-FFPE data set.

FIGURE S5.5: Frequencies of features in classifiers recorded for (A) jointly and (B) separately normalized GSE36681-FFPE data set.

Bibliography

Akkiz, Hikmet (2014). "The Emerging Role of MicroRNAs in Hepatocellular Carcinoma". In: *Euroasian Journal of Hepato-Gastroenterology* 4.1, pp. 45–50. ISSN: 2231-5047. DOI: 10.5005/jp-journals-10018-1095. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5736955/ (visited on 11/19/2022).

Alharbi, Fadi and Aleksandar Vakanski (Jan. 28, 2023). "Machine Learning Methods for Cancer Classification Using Gene Expression Data: A Review". In: *Bioengineering* 10.2, p. 173. ISSN: 2306-5354. DOI: 10.3390/bioengineering10020173. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9952758/ (visited on 06/04/2023).

An, Bolin et al. (Mar. 8, 2023). "Engineered Living Materials For Sustainability". In: *Chemical Reviews* 123.5, pp. 2349–2419. ISSN: 0009-2665, 1520-6890. DOI: 10.1021/acs.chemrev.2c00512. URL: https://pubs.acs.org/doi/10.1021/acs.chemrev.2c00512 (visited on 04/08/2023).

Angelici, Bartolomeo et al. (Dec. 15, 2021). "An AAV gene therapy computes over multiple cellular inputs to enable precise targeting of multifocal hepatocellular carcinoma in mice". In: *Science Translational Medicine* 13.624, eabh4456. ISSN: 1946-6242. DOI: 10.1126/scitranslmed.abh4456.

Bartel, David P. (2009). "MicroRNAs: Target Recognition and Regulatory Functions". In: *Cell* 136.2. ISSN: 00928674. DOI: 10.1016/j.cell.2009.01.002.

Bartoszewicz, Jakub M, Ulrich Genske, and Bernhard Y Renard (Nov. 5, 2021a). "Deep learning-based real-time detection of novel pathogens during sequencing". In: *Briefings in Bioinformatics* 22.6, bbab269. ISSN: 1467-5463, 1477-4054. DOI: 10.1093/bib/bbab269. URL: https://academic.oup.com/bib/article/doi/10.1093/bib/bbab269/6326527 (visited on 06/28/2023).

Bartoszewicz, Jakub M., Anja Seidel, and Bernhard Y. Renard (Mar. 2021b). "Interpretable detection of novel human viruses from genome sequencing data". In: *NAR Genomics and Bioinformatics* 3.lqab004. ISSN: 2631-9268. DOI: 10.1093/nargab/lqab004. URL: https://doi.org/10.1093/nargab/lqab004 (visited on 03/31/2021).

Bartoszewicz, Jakub M. et al. (Jan. 2020). "DeePaC: predicting pathogenic potential of novel DNA with reverse-complement neural networks". In: *Bioinformatics* 36.1, pp. 81–89. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz541. URL: https://doi.org/10.1093/bioinformatics/btz541 (visited on 04/23/2021).

Bartoszewicz, Jakub M et al. (Sept. 16, 2022). "Detecting DNA of novel fungal pathogens using ResNets and a curated fungi-hosts data collection". In: *Bioinformatics* 38 (Supplement_2), pp. ii168–ii174. ISSN: 1367-4803, 1367-4811. DOI: 10.1093/bioinformatics/btac495. URL: https://academic.oup.com/bioinformatics/article/38/Supplement_2/ii168/6702016 (visited on 06/22/2023).

Becker, Katinka et al. (2018). "Designing miRNA-Based Synthetic Cell Classifier Circuits Using Answer Set Programming". In: *Frontiers in Bioengineering and Biotechnology* 6, p. 70. ISSN: 2296-4185. DOI: 10.3389/fbioe.2018.00070. URL: https://www.frontiersin.org/article/10.3389/fbioe.2018.00070/full.

Benenson, Yaakov (Aug. 1, 2009). "RNA-based computation in live cells". In: *Current Opinion in Biotechnology*. Protein technologies / Systems and synthetic biology 20.4, pp. 471–478. ISSN: 0958-1669. DOI: 10.1016/j.copbio.2009.08.002. URL: https://www.sciencedirect.com/science/article/pii/S0958166909000950 (visited on 03/17/2023).

Bergstra, James and Yoshua Bengio (2012). "Random Search for Hyper-Parameter Optimization Yoshua Bengio". In: *Journal of Machine Learning Research* 13, pp. 281–305. URL: http://scikit-learn.sourceforge.net..

Biswas, Surojit et al. (Apr. 2021). "Low-N protein engineering with data-efficient deep learning". In: *Nature Methods* 18.4, pp. 389–396. ISSN: 1548-7105. DOI: 10.1038/s41592-021-01100-y. URL: https://www.nature.com/articles/s41592-021-01100-y (visited on 06/25/2023).

Borriello, Enrico and Bryan C. Daniels (Sept. 1, 2021). "The basis of easy controllability in Boolean networks". In: *Nature Communications* 12.1, p. 5227. ISSN: 2041-1723. DOI: 10.1038/s41467-021-25533-3. URL: https://www.nature.com/articles/s41467-021-25533-3 (visited on 04/17/2023).

Bothfeld, William, Grace Kapov, and Keith E. J. Tyo (July 21, 2017). "A Glucose-Sensing Toggle Switch for Autonomous, High Productivity Genetic Control". In: *ACS Synthetic Biology* 6.7, pp. 1296–1304. ISSN: 2161-5063, 2161-5063. DOI: 10.1021/acssynbio.6b00257. URL: https://pubs.acs.org/doi/10.1021/acssynbio.6b00257 (visited on 03/17/2023).

Bozic, Ivana et al. (June 25, 2013). "Evolutionary dynamics of cancer in response to targeted combination therapy". In: *eLife* 2. Ed. by Carl T Bergstrom, e00747. ISSN: 2050-084X. DOI: 10.7554/eLife.00747. URL: https://doi.org/10.7554/eLife.00747 (visited on 02/20/2022).

Burchard, Julja et al. (2010). "MicroRNA-122 as a regulator of mitochondrial metabolic gene network in hepatocellular carcinoma". In: *Molecular Systems Biology* 6, p. 402. ISSN: 17444292. DOI: 10.1038/msb.2010.58. URL: http://www.ncbi.nlm.nih.gov/pubmed/20739924http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2950084.

Chiu, Tai-Yin and Jie-Hong R. Jiang (Oct. 9, 2017). "Logic Synthesis of Recombinase-Based Genetic Circuits". In: *Scientific Reports* 7.1, p. 12873. ISSN: 2045-2322. DOI: 10.1038/s41598-017-07386-3. URL: https://www.nature.com/articles/s41598-017-07386-3 (visited on 06/26/2023).

Cho, Jang Hwan, James J. Collins, and Wilson W. Wong (May 31, 2018). "Universal Chimeric Antigen Receptors for Multiplexed and Logical Control of T Cell Responses". In: *Cell* 173.6, 1426–1438.e11. ISSN: 1097-4172. DOI: 10.1016/j.cell.2018.03.038.

Cifuentes-Fontanals, Laura, Elisa Tonello, and Heike Siebert (2022). "Control in Boolean Networks With Model Checking". In: *Frontiers in Applied Mathematics and Statistics* 8. ISSN: 2297-4687. URL: https://www.frontiersin.org/articles/10.3389/fams.2022.838546 (visited on 04/17/2023).

Courbet, Alexis et al. (May 27, 2015). "Detection of pathological biomarkers in human clinical samples via amplifying genetic switches and logic gates". In: *Science Translational Medicine* 7.289, 289ra83. ISSN: 1946-6242. DOI: 10.1126/scitranslmed.aaa3601.

Creighton, Chad J. (Jan. 2018). "Making Use of Cancer Genomic Databases". en. In: *Current Protocols in Molecular Biology* 121.1. ISSN: 1934-3639, 1934-3647. DOI: 10.1002/cpmb.49. URL: https://onlinelibrary.wiley.com/doi/10.1002/cpmb.49 (visited on 07/27/2021).

Cubillos-Ruiz, Andres et al. (Dec. 2021). "Engineering living therapeutics with synthetic biology". In: *Nature Reviews Drug Discovery* 20.12, pp. 941–960. ISSN: 1474-1784. DOI: 10.1038/s41573-021-00285-3. URL: https://www.nature.com/articles/s41573-021-00285-3 (visited on 03/15/2023).

Dagogo-Jack, Ibiayi and Alice T. Shaw (Feb. 2018). "Tumour heterogeneity and resistance to cancer therapies". en. In: *Nature Reviews Clinical Oncology* 15.2, pp. 81–94. ISSN: 1759-4782. DOI: 10.1038/nrclinonc.2017.166. URL: https://www.nature.com/articles/nrclinonc.2017.166 (visited on 07/12/2021).

Daniel, Ramiz et al. (May 2013). "Synthetic analog computation in living cells". In: *Nature* 497.7451, pp. 619–623. ISSN: 1476-4687. DOI: 10.1038/nature12148. URL: https://www.nature.com/articles/nature12148 (visited on 03/14/2023).

Dastor, Margaux et al. (Feb. 16, 2018). "A Workflow for *In Vivo* Evaluation of Candidate Inputs and Outputs for Cell Classifier Gene Circuits". In: *ACS Synthetic Biology* 7.2, pp. 474–489. ISSN: 2161-5063, 2161-5063. DOI: 10.1021/acssynbio.7b00303. URL: https://pubs.acs.org/doi/10.1021/acssynbio.7b00303 (visited on 04/10/2023).

Didovyk, Andriy et al. (2015). "Distributed classifier based on genetically engineered bacterial cell cultures". In: *ACS Synthetic Biology* 4.1, pp. 27–82. ISSN: 21615063. DOI: 10.1021/sb500235p.

Dillies, M.-A. et al. (2013). "A comprehensive evaluation of normalization methods for Illumina high-throughput RNA sequencing data analysis". In: *Briefings in Bioinformatics* 14.6, pp. 671–683. ISSN: 1467-5463. DOI: 10.1093/bib/bbs046. URL: https://academic.oup.com/bib/article-lookup/doi/10.1093/bib/bbs046.

Ding, Yanling et al. (2017). "miR-145 inhibits proliferation and migration of breast cancer cells by directly or indirectly regulating TGF-β1 expression". In: *International Journal of Oncology* 50.5, pp. 1701–1710. ISSN: 1019-6439. DOI: 10.3892/ijo.2017.3945.

Doudna, Jennifer A. and Emmanuelle Charpentier (Nov. 28, 2014). "The new frontier of genome engineering with CRISPR-Cas9". In: *Science* 346.6213, p. 1258096. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1258096. URL: https://www.science.org/doi/10.1126/science.1258096 (visited on 04/09/2023).

Dupont, Camille Amandine et al. (2021). "Druggable genome and precision medicine in cancer: current challenges". en. In: *The FEBS Journal* n/a.n/a. ISSN: 1742-4658. DOI: 10.1111/febs.15788. URL: https://febs.onlinelibrary.wiley.com/doi/abs/10.1111/febs.15788 (visited on 07/26/2021).

Edgar, R. (2002). "Gene Expression Omnibus: NCBI gene expression and hybridization array data repository". In: *Nucleic Acids Research* 30.1, pp. 207–210. ISSN: 1362-4962. DOI: 10.1093/nar/30.1.207.

El Karoui, Meriem, Monica Hoyos-Flight, and Liz Fletcher (2019). "Future Trends in Synthetic Biology—A Report". In: *Frontiers in Bioengineering and Biotechnology* 7, p. 175. ISSN: 2296-4185. DOI: 10.3389/fbioe.2019.00175. URL: https://www.frontiersin.org/article/10.3389/fbioe.2019.00175 (visited on 12/11/2021).

Endy, Drew (Nov. 2005). "Foundations for engineering biology". In: *Nature* 438.7067, pp. 449–453. ISSN: 1476-4687. DOI: 10.1038/nature04342. URL: https://www.nature.com/articles/nature04342 (visited on 03/18/2023).

Falkner, Andreas et al. (Aug. 1, 2018). "Industrial Applications of Answer Set Programming". In: *KI - Künstliche Intelligenz* 32.2, pp. 165–176. ISSN: 1610-1987. DOI: 10.1007/s13218-018-0548-6. URL: https://doi.org/10.1007/s13218-018-0548-6 (visited on 06/18/2023).

Farazi, T. A. et al. (2011). "MicroRNA Sequence and Expression Analysis in Breast Tumors by Deep Sequencing". In: *Cancer Research* 71.13, pp. 4443–4453. ISSN: 0008-5472. DOI: 10.1158/0008-5472.CAN-11-0608.

Febbraro, Onofrio, Kristian Reale, and Francesco Ricca (2011). "ASPIDE: Integrated Development Environment for Answer Set Programming". In: DOI: 10.1007/978-3-642-20895-9_37.

Fornari, F. et al. (Sept. 2008). "MiR-221 controls CDKN1C/p57 and CDKN1B/p27 expression in human hepatocellular carcinoma". In: *Oncogene* 27.43, pp. 5651–5661. ISSN: 1476-5594. DOI: 10.1038/onc.2008.178. URL: https://www.nature.com/articles/onc2008178 (visited on 11/19/2022).

Gallo, C.A. et al. (2016a). "Discretization of gene expression data revised." In: *Briefings in Bioinformatics* 17.5, pp. 758–770. DOI: 10.1158/0008-5472.CAN-11-0608. URL: https://academic.oup.com/bib/article/17/5/758/2261412.

Gallo, Cristian A. et al. (2016b). "Discretization of gene expression data revised". In: *Briefings in Bioinformatics* 17.5, pp. 758–770. ISSN: 14774054. DOI: 10.1093/bib/bbv074.

Gebser, Martin, Benjamin Kaufmann, and Torsten Schaub (2012). "Conflict-driven answer set solving: From theory to practice". In: *Artificial Intelligence* 187-188, pp. 52–89. ISSN: 0004-3702. DOI: 10.1016/J.ARTINT.2012.04.001. URL: https://www.sciencedirect.com/science/article/pii/S0004370212000409.

Gebser, Martin et al. (Oct. 2010). "The BioASP Library: ASP Solutions for Systems Biology". In: *2010 22nd IEEE International Conference on Tools with Artificial Intelligence*. 2010 22nd IEEE International Conference on Tools with Artificial Intelligence. Vol. 1. ISSN: 2375-0197, pp. 383–389. DOI: 10.1109/ICTAI.2010.62.

Gorochowski, Thomas E. et al. (2021). "Editorial: Computer-Aided Biodesign Across Scales". In: *Frontiers in Bioengineering and Biotechnology* 9. ISSN: 2296-4185. URL: https://www.frontiersin.org/articles/10.3389/fbioe.2021.700418 (visited on 03/15/2023).

Grupp, Stephan A. et al. (Apr. 2013). "Chimeric Antigen Receptor–Modified T Cells for Acute Lymphoid Leukemia". en. In: *New England Journal of Medicine* 368.16, pp. 1509–1518. ISSN: 0028-4793, 1533-4406. DOI: 10.1056/NEJMoa1215134. URL: http://www.nejm.org/doi/10.1056/NEJMoa1215134 (visited on 07/23/2021).

Guiziou, Sarah, Pauline Mayonove, and Jerome Bonnet (Jan. 28, 2019). "Hierarchical composition of reliable recombinase logic devices". In: *Nature Communications* 10.1, p. 456. ISSN: 2041-1723. DOI: 10.1038/s41467-019-08391-y. URL: https://www.nature.com/articles/s41467-019-08391-y (visited on 04/08/2023).

Guiziou, Sarah et al. (May 18, 2018). "An Automated Design Framework for Multicellular Recombinase Logic". In: *ACS Synthetic Biology* 7.5, pp. 1406–1412. ISSN: 2161-5063. DOI: 10.1021/acssynbio.8b00016. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5962929/ (visited on 03/17/2023).

Guziolowski, Carito et al. (2013). "Exhaustively characterizing feasible logic models of a signaling network using Answer Set Programming". In: *Bioinformatics* 29.18. ISSN: 1460-2059. DOI: 10.1093/bioinformatics/btt393.

Haixiang, Guo et al. (2017). "Learning from class-imbalanced data: Review of methods and applications". In: *Expert Systems with Applications* 73, pp. 220–239. ISSN: 0957-4174. DOI: 10.1016/J.ESWA.2016.12.035.

Hashimoto, Yutaka, Yoshimitsu Akiyama, and Yasuhito Yuasa (2013). "Multiple-to-Multiple Relationships between MicroRNAs and Target Genes in Gastric Cancer". In: *PLoS ONE* 8.5. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0062589.

Hibstu, Zigale et al. (Oct. 6, 2022). "Phage Therapy: A Different Approach to Fight Bacterial Infections". In: *Biologics : Targets & Therapy* 16, pp. 173–186. ISSN: 1177-5475. DOI: 10.2147/BTT.S381237. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9550173/ (visited on 06/22/2023).

Hiscock, Tom W. (Apr. 27, 2019). "Adapting machine-learning algorithms to design gene circuits". In: *BMC Bioinformatics* 20.1, p. 214. ISSN: 1471-2105. DOI: 10.1186/s12859-019-2788-3. URL: https://doi.org/10.1186/s12859-019-2788-3 (visited on 03/15/2023).

Ho, Joanne M. L. and Matthew R. Bennett (Apr. 13, 2018). "Improved memory devices for synthetic cells". In: *Science* 360.6385, pp. 150–151. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aat3236. URL: https://www.science.org/doi/10.1126/science.aat3236 (visited on 04/08/2023).

Hsu, Patrick D., Eric S. Lander, and Feng Zhang (June 5, 2014). "Development and Applications of CRISPR-Cas9 for Genome Engineering". In: *Cell* 157.6, pp. 1262–1278. ISSN: 0092-8674. DOI: 10.1016/j.cell.2014.05.010. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4343198/ (visited on 04/09/2023).

Hérisson, Joan et al. (Aug. 29, 2022). "The automated Galaxy-SynBioCAD pipeline for synthetic biology design and engineering". In: *Nature Communications* 13.1, p. 5082. ISSN: 2041-1723. DOI: 10.1038/s41467-022-32661-x. URL: https://www.nature.com/articles/s41467-022-32661-x (visited on 03/15/2023).

Iorio, Marilena V and Carlo M Croce (2012). "MicroRNA dysregulation in cancer: diagnostics, monitoring and therapeutics. A comprehensive review." In: *EMBO molecular medicine* 4.3, pp. 143–59. ISSN: 1757-4684. DOI: 10.1002/emmm.201100209.

Jang, Jin Sung et al. (2012). "Increased miR-708 expression in NSCLC and its association with poor survival in lung adenocarcinoma from never smokers". In: *Clinical Cancer Research* 18.13, pp. 3658–3667. ISSN: 10780432. DOI: 10.1158/1078-0432.CCR-11-2857. URL: http://www.ncbi.nlm.nih.gov/pubmed/22573352http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3616503.

Kanakov, Oleg et al. (2015). "Multi-Input Distributed Classifiers for Synthetic Genetic Circuits". In: *PLOS ONE* 10.5. Ed. by Mark Isalan, e0125144. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0125144. URL: https://dx.plos.org/10.1371/journal.pone.0125144.

Katoch, Sourabh, Sumit Singh Chauhan, and Vijay Kumar (2021). "A review on genetic algorithm: past, present, and future". In: *Multimedia Tools and Applications* 80.5. ISSN: 1380-7501. DOI: 10.1007/s11042-020-10139-6.

Kaufmann, Benjamin et al. (2016). "Grounding and Solving in Answer Set Programming". In: *AI Magazine* 37.3, pp. 25–32. DOI: 10.1609/aimag.v37i3.2672. URL: https://ojs.aaai.org/index.php/aimagazine/article/view/2672.

Kazimipour, Borhan, Xiaodong Li, and A. K. Qin (July 2014). "A review of population initialization techniques for evolutionary algorithms". In: *2014 IEEE Congress on Evolutionary Computation (CEC)*. 2014 IEEE Congress on Evolutionary Computation (CEC). Beijing, China: IEEE, pp. 2585–2592. ISBN: 9781479914883 9781479966264. DOI: 10.1109/CEC.2014.6900618. URL: https://ieeexplore.ieee.org/document/6900618 (visited on 06/07/2021).

Kitano, Shohei et al. (2023). "Synthetic biology: Learning the way toward high-precision biological design". In: *PLOS Biology* 21.4, e3002116. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.3002116. URL: https://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.3002116 (visited on 06/25/2023).

Kortright, Kaitlyn E. et al. (Feb. 13, 2019). "Phage Therapy: A Renewed Approach to Combat Antibiotic-Resistant Bacteria". In: *Cell Host & Microbe* 25.2, pp. 219–232. ISSN: 1934-6069. DOI: 10.1016/j.chom.2019.01.014.

Lan, Huiyin et al. (2015). "MicroRNAs as potential biomarkers in cancer: opportunities and challenges." In: *BioMed research international* 2015, p. 125094. ISSN: 2314-6141. DOI: 10.1155/2015/125094.

Lapique, Nicolas and Yaakov Benenson (Dec. 2014). "Digital switching in a biosensor circuit via programmable timing of gene availability". In: *Nature chemical biology* 10.12, pp. 1020–1027. ISSN: 1552-4450. DOI: 10.1038/nchembio.1680. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4232471/ (visited on 02/15/2017).

Li, Hua et al. (July 31, 2019). "miR-450b-5p loss mediated KIF26B activation promoted hepatocellular carcinoma progression by activating PI3K/AKT pathway". In: *Cancer Cell International* 19, p. 205. ISSN: 1475-2867. DOI: 10.1186/s12935-019-0923-x. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6670205/ (visited on 11/18/2022).

Li, Wenxi et al. (2008). "Diagnostic and prognostic implications of microRNAs in human hepatocellular carcinoma". In: *International Journal of Cancer* 123.7, pp. 1616–1622. ISSN: 00207136. DOI: 10.1002/ijc.23693. URL: http://www.ncbi.nlm.nih.gov/pubmed/18649363.

Li, Yongqing et al. (2015). "Associations of miR-146a and miR-146b expression and breast cancer in very young women". In: *Cancer Biomarkers* 15.6, pp. 881–887. ISSN: 15740153. DOI: 10.3233/CBM-150532.

Lin, Ching-Wen et al. (May 21, 2013). "MicroRNA-135b promotes lung cancer metastasis by regulating multiple targets in the Hippo pathway and LZTS1". In: *Nature Communications* 4.1, p. 1877. ISSN: 2041-1723. DOI: 10.1038/ncomms2876. URL: https://www.nature.com/articles/ncomms2876 (visited on 11/19/2022).

Liu, Yanfeng, Aimin Zhou, and Hu Zhang (2018). "Termination detection strategies in evolutionary algorithms". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM. ISBN: 9781450356183. DOI: 10.1145/3205455.3205466.

Lopez-Rincon, Alejandro et al. (Sept. 18, 2019). "Automatic discovery of 100-miRNA signature for cancer classification using ensemble feature selection". In: *BMC Bioinformatics* 20.1, p. 480. ISSN: 1471-2105. DOI: 10.1186/s12859-019-3050-8. URL: https://doi.org/10.1186/s12859-019-3050-8 (visited on 06/04/2023).

Ludwig, Nicole et al. (2016). "Distribution of miRNA expression across human tissues." In: *Nucleic acids research* 44.8, pp. 3865–77. ISSN: 1362-4962. DOI: 10.1093/nar/gkw116. URL: http://www.ncbi.nlm.nih.gov/pubmed/26921406http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4856985.

Ma, Jincai et al. (Aug. 2019). "LncRNA PAPAS promotes hepatocellular carcinoma by interacting with miR-188-5p". In: *Journal of Cellular Biochemistry* 120.8, pp. 13494–13500. ISSN: 0730-2312, 1097-4644. DOI: 10.1002/jcb.28623. URL: https://onlinelibrary.wiley.com/doi/10.1002/jcb.28623 (visited on 11/18/2022).

Manning, Timmy, Roy D Sleator, and Paul Walsh (2013). "Naturally selecting solutions: the use of genetic algorithms in bioinformatics." In: *Bioengineered* 4.5, pp. 266–78. ISSN: 2165-5987. DOI: 10.4161/bioe.23041.

Mao, Long-fei et al. (2022). "Design, Synthesis, and Antitumor Activity of Erlotinib Derivatives". In: *Frontiers in Pharmacology* 13. ISSN: 1663-9812. URL: https://www.frontiersin.org/articles/10.3389/fphar.2022.849364 (visited on 06/21/2023).

Marusyk, Andriy, Vanessa Almendro, and Kornelia Polyak (May 2012). "Intra-tumour heterogeneity: a looking glass for cancer?" en. In: *Nature Reviews Cancer* 12.5, pp. 323–334. ISSN: 1474-1768. DOI: 10.1038/nrc3261. URL: https://www.nature.com/articles/nrc3261 (visited on 07/18/2021).

McCall, John (2005). "Genetic algorithms for modelling and optimisation". In: *Journal of Computational and Applied Mathematics* 184.1, pp. 205–222. ISSN: 0377-0427. DOI: 10.1016/J.CAM.2004.07.034.

McInnes, Leland, John Healy, and James Melville (Sept. 2020). "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction". arXiv:1802.03426 [cs, stat] type: article. URL: http://arxiv.org/abs/1802.03426 (visited on 07/24/2022).

McLaughlin, James Alastair et al. (2020). "The Synthetic Biology Open Language (SBOL) Version 3: Simplified Data Exchange for Bioengineering". In: *Frontiers in Bioengineering and Biotechnology* 8. ISSN: 2296-4185. URL: https://www.frontiersin.org/articles/10.3389/fbioe.2020.01009 (visited on 03/18/2023).

Meng, Fankang and Tom Ellis (Oct. 14, 2020). "The second decade of synthetic biology: 2010–2020". In: *Nature Communications* 11, p. 5174. ISSN: 2041-1723. DOI: 10.1038/s41467-020-19092-2. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7560693/ (visited on 12/11/2021).

Miki, Kenji et al. (2015). "Efficient Detection and Purification of Cell Populations Using Synthetic MicroRNA Switches". In: *Cell Stem Cell* 16.6, pp. 699–711. ISSN: 18759777. DOI: 10.1016/j.stem.2015.04.005.

Mitchell, Melanie (1996). "Chapter 1: Genetic Algorithms: An Overview". In: *An Introduction to Genetic Algorithms*. Complex Adaptive Systems. MIT Press.

Mohammadi, Pejman, Niko Beerenwinkel, and Yaakov Benenson (2017). "Automated Design of Synthetic Cell Classifier Circuits Using a Two-Step Optimization Strategy". In: *Cell Systems* 4.2, 207–218.e14. ISSN: 24054720. DOI: 10.1016/j.cels.2017.01.003.

Moon, T.S. et al. (2012). "Genetic programs constructed from layered logic gates in single cells." In: *Nature* 491.7423, pp. 249–253. DOI: 10.1038/nature11516. URL: https://www.nature.com/articles/nature11516.

Ng, Enders K. O. et al. (2014). "MicroRNA-143 is downregulated in breast cancer and regulates DNA methyltransferases 3A in breast cancer cells". In: *Tumor Biology* 35.3, pp. 2591–2598. ISSN: 1010-4283. DOI: 10.1007/s13277-013-1341-7.

Nielsen, A. A. K. et al. (Apr. 1, 2016). "Genetic circuit design automation". In: *Science* 352.6281, aac7341–aac7341. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aac7341. URL: https://www.sciencemag.org/lookup/doi/10.1126/science.aac7341 (visited on 03/18/2023).

Nissim, Lior et al. (Nov. 16, 2017). "Synthetic RNA-Based Immunomodulatory Gene Circuits for Cancer Immunotherapy". In: *Cell* 171.5, 1138–1150.e15. ISSN: 0092-8674. DOI: 10.1016/j.cell.2017.09.049. URL: https://www.sciencedirect.com/science/article/pii/S0092867417311431 (visited on 04/15/2023).

Nowicka, Melania and Heike Siebert (2019). "Designing Distributed Cell Classifier Circuits Using a Genetic Algorithm". In: *Lecture Notes in Computer Science*. Vol. 11773 LNBI. Springer, pp. 96–119. ISBN: 9783030313036. DOI: 10.1007/978-3-030-31304-3_6.

— (2020). "A framework for designing miRNA-based distributed cell classifier circuits". In: *bioRxiv*. DOI: 10.1101/2020.05.13.092908. eprint: https://www.biorxiv.org/content/early/2020/05/15/2020.05.13.092908.full.pdf. URL: https://www.biorxiv.org/content/early/2020/05/15/2020.05.13.092908.

Oishi, Tomoya et al. (Sept. 8, 2022). "Efficacy of HSV-TK/GCV system suicide gene therapy using SHED expressing modified HSV-TK against lung cancer brain metastases". In: *Molecular Therapy - Methods & Clinical Development* 26, pp. 253–265. ISSN: 2329-0501. DOI: 10.1016/j.omtm.2022.07.001. URL: https://www.

cell.com/molecular-therapy-family/methods/abstract/S2329-0501(22)00093-6 (visited on 04/15/2023).

Opitz, D. and R. Maclin (Aug. 1, 1999a). "Popular Ensemble Methods: An Empirical Study". In: *Journal of Artificial Intelligence Research* 11, pp. 169–198. ISSN: 1076-9757. DOI: 10.1613/jair.614. URL: https://jair.org/index.php/jair/article/view/10239 (visited on 06/16/2021).

— (Aug. 1, 1999b). "Popular Ensemble Methods: An Empirical Study". In: *Journal of Artificial Intelligence Research* 11, pp. 169–198. ISSN: 1076-9757. DOI: 10.1613/jair.614. URL: https://jair.org/index.php/jair/article/view/10239 (visited on 06/26/2023).

Palu, Alessandro Dal et al. (2018). "Exploring life: answer set programming in bioinformatics". In: *Declarative Logic Programming: Theory, Systems, and Applications*. ACM. DOI: 10.1145/3191315.3191323.

Pan, Y., J. Zhang, and L. Shen (2016). "miR-144 functions as a tumor suppressor in breast cancer through inhibiting ZEB1/2-mediated epithelial mesenchymal transition process." In: *OncoTargets and Therapy* 9, pp. 6247–6255. DOI: 10.2147/OTT.S103650. URL: https://www.dovepress.com/mir-144-functions-as-a-tumor-suppressor-in-breast-cancer-through-inhib-peer-reviewed-article-OTT.

Parca, Luca et al. (Oct. 23, 2019). "Modeling cancer drug response through drug-specific informative genes". In: *Scientific Reports* 9.1, p. 15222. ISSN: 2045-2322. DOI: 10.1038/s41598-019-50720-0. URL: https://www.nature.com/articles/s41598-019-50720-0 (visited on 06/28/2023).

Pineau, Pascal et al. (Jan. 5, 2010). "miR-221 overexpression contributes to liver tumorigenesis". In: *Proceedings of the National Academy of Sciences* 107.1, pp. 264–269. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.0907904107. URL: https://pnas.org/doi/full/10.1073/pnas.0907904107 (visited on 11/19/2022).

Plahar, Hector A. et al. (Oct. 15, 2021). "BioParts—A Biological Parts Search Portal and Updates to the ICE Parts Registry Software Platform". In: *ACS Synthetic Biology* 10.10, pp. 2649–2660. ISSN: 2161-5063, 2161-5063. DOI: 10.1021/acssynbio.1c00263. URL: https://pubs.acs.org/doi/10.1021/acssynbio.1c00263 (visited on 06/28/2023).

Polikar, R. (2006). "Ensemble based systems in decision making". In: *IEEE Circuits and Systems Magazine* 6.3, pp. 21–45. ISSN: 1531-636X. DOI: 10.1109/MCAS.2006.1688199. URL: http://ieeexplore.ieee.org/document/1688199/ (visited on 06/16/2021).

Prochazka, Laura et al. (Nov. 11, 2022). "Synthetic gene circuits for cell state detection and protein tuning in human pluripotent stem cells". In: *Molecular Systems Biology* 18.11, e10886. ISSN: 1744-4292. DOI: 10.15252/msb.202110886. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9650275/ (visited on 06/25/2023).

Ramola, Rashika, Shantanu Jain, and Predrag Radivojac (2019). "Estimating classification accuracy in positive-unlabeled learning: characterization and correction strategies." In: *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing* 24, pp. 124–135. ISSN: 2335-6936.

Robinson, Mark D., Davis J. McCarthy, and Gordon K. Smyth (2009). "edgeR: A Bioconductor package for differential expression analysis of digital gene expression data". In: *Bioinformatics* 26.1, pp. 139–140. ISSN: 14602059. DOI: 10.1093/bioinformatics/btp616.

Roell, Marc-Sven and Matias D Zurbriggen (Feb. 1, 2020). "The impact of synthetic biology for future agriculture and nutrition". In: *Current Opinion in Biotechnology* 61, pp. 102–109. ISSN: 0958-1669. DOI: 10.1016/j.copbio.2019.10.004. URL:

https://www.sciencedirect.com/science/article/pii/S095816691930103X (visited on 03/15/2023).

Rokach, Lior (Feb. 2010). "Ensemble-based classifiers". In: *Artificial Intelligence Review* 33.1, pp. 1–39. ISSN: 0269-2821, 1573-7462. DOI: 10.1007/s10462-009-9124-7. URL: http://link.springer.com/10.1007/s10462-009-9124-7 (visited on 06/16/2021).

Roscigno, G. et al. (2017). "MiR-24 induces chemotherapy resistance and hypoxic advantage in breast cancer." In: *Oncotarget* 8.12, pp. 19507–19521. DOI: 10.3389/fnins.2013.12345. URL: http://www.oncotarget.com/index.php?journal=oncotarget&page=article&op=view&path[]=14470&path[]=46158.

Rubinfeld, Bonnee et al. (Feb. 2006). "Identification and immunotherapeutic targeting of antigens induced by chemotherapy". In: *Nature Biotechnology* 24.2, pp. 205–209. ISSN: 1546-1696. DOI: 10.1038/nbt1185. URL: https://www.nature.com/articles/nbt1185 (visited on 02/20/2022).

Russell, Daniel A. and Graham F. Hatfull (Mar. 2017). "PhagesDB: the actinobacteriophage database". In: *Bioinformatics* 33.5, pp. 784–786. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btw711. URL: https://doi.org/10.1093/bioinformatics/btw711 (visited on 04/14/2021).

Sarver, Aaron L. et al. (Dec. 2018). "OMCD: OncomiR Cancer Database". In: *BMC Cancer* 18.1, p. 1223. ISSN: 1471-2407. DOI: 10.1186/s12885-018-5085-z. URL: https://doi.org/10.1186/s12885-018-5085-z (visited on 08/01/2021).

Sayeg, Marianna K. et al. (July 17, 2015). "Rationally Designed MicroRNA-Based Genetic Classifiers Target Specific Neurons in the Brain". In: *ACS Synthetic Biology* 4.7, pp. 788–795. ISSN: 2161-5063, 2161-5063. DOI: 10.1021/acssynbio.5b00040. URL: https://pubs.acs.org/doi/10.1021/acssynbio.5b00040 (visited on 06/25/2023).

Schmidt, Tiziana Julia Nadjeschda et al. (Jan. 1, 2023). "CRISPR/Cas9 in the era of nanomedicine and synthetic biology". In: *Drug Discovery Today* 28.1, p. 103375. ISSN: 1359-6446. DOI: 10.1016/j.drudis.2022.103375. URL: https://www.sciencedirect.com/science/article/pii/S1359644622003683 (visited on 04/09/2023).

Schätzle, Lisa-Katrin, Ali Hadizadeh Esfahani, and Andreas Schuppert (Apr. 20, 2020). "Methodological challenges in translational drug response modeling in cancer: A systematic analysis with FORESEE". In: *PLoS Computational Biology* 16.4, e1007803. ISSN: 1553-734X. DOI: 10.1371/journal.pcbi.1007803. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7192505/ (visited on 06/28/2023).

Setty, Y. et al. (June 24, 2003). "Detailed map of a cis-regulatory input function". In: *Proceedings of the National Academy of Sciences* 100.13, pp. 7702–7707. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1230759100. URL: https://pnas.org/doi/full/10.1073/pnas.1230759100 (visited on 03/17/2023).

Shukla, Anupriya, Hari Mohan Pandey, and Deepti Mehrotra (2015). "Comparative review of selection techniques in genetic algorithm". In: *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*. IEEE, pp. 515–519. ISBN: 978-1-4799-8433-6. DOI: 10.1109/ABLAZE.2015.7154916.

Smith, Robert W., Bob van Sluijs, and Christian Fleck (2017). "Designing synthetic networks in silico: a generalised evolutionary algorithm approach". In: *BMC Systems Biology* 11.1, p. 118. ISSN: 1752-0509. DOI: 10.1186/s12918-017-0499-9.

Soneson, Charlotte (2014). "compcodeR - An R package for benchmarking differential expression methods for RNA-seq data". In: *Bioinformatics* 30.17, pp. 2517–

2518. ISSN: 14602059. DOI: 10.1093/bioinformatics/btu324. URL: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu324.

Soneson, Charlotte and Mauro Delorenzi (2013). "A comparison of methods for differential expression analysis of RNA-seq data". In: *BMC Bioinformatics* 14.1, p. 91. ISSN: 1471-2105. DOI: 10.1186/1471-2105-14-91. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-14-91.

Spellberg, Brad (June 27, 2014). "The future of antibiotics". In: *Critical Care* 18.3, p. 228. ISSN: 1364-8535. DOI: 10.1186/cc13948. URL: https://doi.org/10.1186/cc13948 (visited on 06/22/2023).

Suzuki, Rui et al. (Oct. 22, 2018). "miR-182 and miR-183 Promote Cell Proliferation and Invasion by Targeting FOXO1 in Mesothelioma". In: *Frontiers in Oncology* 8, p. 446. ISSN: 2234-943X. DOI: 10.3389/fonc.2018.00446. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6204457/ (visited on 11/19/2022).

Sánchez-Cid, Lourdes et al. (2017). "MicroRNA-200, associated with metastatic breast cancer, promotes traits of mammary luminal progenitor cells." In: *Oncotarget* 8.48, pp. 83384–83406. ISSN: 1949-2553. DOI: 10.18632/oncotarget.20698.

"The global challenge of cancer" (Jan. 2020). en. In: *Nature Cancer* 1.1, pp. 1–2. ISSN: 2662-1347. DOI: 10.1038/s43018-019-0023-9. URL: https://www.nature.com/articles/s43018-019-0023-9 (visited on 07/17/2021).

Thompson, Tosin (Jan. 31, 2022). "The staggering death toll of drug-resistant bacteria". In: *Nature*. DOI: 10.1038/d41586-022-00228-x. URL: https://www.nature.com/articles/d41586-022-00228-x (visited on 06/22/2023).

Umesh, P. et al. (Dec. 2010). "Programming languages for synthetic biology". In: *Systems and Synthetic Biology* 4.4, pp. 265–269. ISSN: 1872-5325. DOI: 10.1007/s11693-011-9070-y. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3065592/ (visited on 03/18/2023).

Vasić, Marko, David Soloveichik, and Sarfraz Khurshid (June 1, 2020). "CRN++: Molecular programming language". In: *Natural Computing* 19.2, pp. 391–407. ISSN: 1572-9796. DOI: 10.1007/s11047-019-09775-1. URL: https://doi.org/10.1007/s11047-019-09775-1 (visited on 03/18/2023).

Voigt, Christopher A. (Dec. 11, 2020). "Synthetic biology 2020–2030: six commercially-available products that are changing our world". In: *Nature Communications* 11.1, p. 6379. ISSN: 2041-1723. DOI: 10.1038/s41467-020-20122-2. URL: https://www.nature.com/articles/s41467-020-20122-2 (visited on 03/14/2023).

Wang, Hong-Qiang, Gao-Jian Jing, and Chunhou Zheng (2014). "Biology-constrained gene expression discretization for cancer classification". In: *Neurocomputing* 145, pp. 30–36. ISSN: 0925-2312. DOI: 10.1016/J.NEUCOM.2014.04.064. URL: https://www.sciencedirect.com/science/article/pii/S0925231214008480.

Wang, Huimin et al. (Mar. 1, 2019). "MiR-183-5p is required for non-small cell lung cancer progression by repressing PTEN". In: *Biomedicine & Pharmacotherapy* 111, pp. 1103–1111. ISSN: 0753-3322. DOI: 10.1016/j.biopha.2018.12.115. URL: https://www.sciencedirect.com/science/article/pii/S0753332218358840 (visited on 11/19/2022).

Wang, Yuhua, Brenda F. Canine, and Arash Hatefi (Apr. 2011). "HSV-TK/GCV Cancer Suicide Gene Therapy by a Designed Recombinant Multifunctional Vector". In: *Nanomedicine : nanotechnology, biology, and medicine* 7.2, pp. 193–200. ISSN: 1549-9634. DOI: 10.1016/j.nano.2010.08.003. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3024463/ (visited on 04/15/2023).

Ward, Richard A. et al. (Mar. 2021). "Challenges and Opportunities in Cancer Drug Resistance". In: *Chemical Reviews* 121.6, pp. 3297–3351. ISSN: 0009-2665. DOI: 10.

1021/acs.chemrev.0c00383. URL: https://doi.org/10.1021/acs.chemrev.0c00383 (visited on 07/17/2021).

Whalen, Sean et al. (Mar. 2022). "Navigating the pitfalls of applying machine learning in genomics". In: *Nature Reviews Genetics* 23.3, pp. 169–181. ISSN: 1471-0064. DOI: 10.1038/s41576-021-00434-9. URL: https://www.nature.com/articles/s41576-021-00434-9 (visited on 05/18/2022).

WHO report on cancer: setting priorities, investing wisely and providing care for all (2020). Tech. rep. Geneva: World Health Organization.

Xie, Z. et al. (2011). "Multi-Input RNAi-Based Logic Circuit for Identification of Specific Cancer Cells". In: *Science* 333.6047, pp. 1307–1311. ISSN: 0036-8075. DOI: 10.1126/science.1205527.

Xu, Xiaoyang et al. (Apr. 1, 2015). "Cancer nanomedicine: from targeted delivery to combination therapy". In: *Trends in Molecular Medicine* 21.4, pp. 223–232. ISSN: 1471-4914. DOI: 10.1016/j.molmed.2015.01.001. URL: https://www.sciencedirect.com/science/article/pii/S1471491415000027 (visited on 02/20/2022).

Yang, Kun, Jianzhong Li, and Hong Gao (2006). "The impact of sample imbalance on identifying differentially expressed genes." In: *BMC bioinformatics* 7 Suppl 4.Suppl 4, S8. ISSN: 1471-2105. DOI: 10.1186/1471-2105-7-S4-S8.

Yang, Wanjuan et al. (Nov. 22, 2012). "Genomics of Drug Sensitivity in Cancer (GDSC): a resource for therapeutic biomarker discovery in cancer cells". In: *Nucleic Acids Research* 41 (D1), pp. D955–D961. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gks1111. URL: http://academic.oup.com/nar/article/41/D1/D955/1059448/Genomics-of-Drug-Sensitivity-in-Cancer-GDSC-a (visited on 06/22/2023).

Yang, Yan et al. (Aug. 22, 2017a). "MicroRNA-218 functions as a tumor suppressor in lung cancer by targeting IL-6/STAT3 and negatively correlates with poor prognosis". In: *Molecular Cancer* 16.1, p. 141. ISSN: 1476-4598. DOI: 10.1186/s12943-017-0710-z. URL: https://doi.org/10.1186/s12943-017-0710-z (visited on 11/19/2022).

Yang, Zhen et al. (2017b). "dbDEMC 2.0: updated database of differentially expressed miRNAs in human cancers". In: *Nucleic Acids Research* 45.D1, pp. D812–D818. ISSN: 0305-1048. DOI: 10.1093/nar/gkw1079. URL: http://www.ncbi.nlm.nih.gov/pubmed/27899556http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5210560https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkw1079.

Yeh, Andy Hsien-Wei et al. (Feb. 2023). "De novo design of luciferases using deep learning". In: *Nature* 614.7949, pp. 774–780. ISSN: 1476-4687. DOI: 10.1038/s41586-023-05696-3. URL: https://www.nature.com/articles/s41586-023-05696-3 (visited on 04/09/2023).

Zhang, Tianze et al. (Jan. 19, 2016). "Downregulation of miR-522 suppresses proliferation and metastasis of non-small cell lung cancer cells by directly targeting DENN/MADD domain containing 2D". In: *Scientific Reports* 6.1, p. 19346. ISSN: 2045-2322. DOI: 10.1038/srep19346. URL: https://www.nature.com/articles/srep19346 (visited on 11/19/2022).

Zhang, W. et al. (July 2017). "Piccolo mediates EGFR signaling and acts as a prognostic biomarker in esophageal squamous cell carcinoma". In: *Oncogene* 36.27, pp. 3890–3902. ISSN: 1476-5594. DOI: 10.1038/onc.2017.15. URL: https://www.nature.com/articles/onc201715 (visited on 06/21/2023).

Zhang, Xiping, Hongjian Yang, and Ruiping Zhang (Sept. 2019). "Challenges and future of precision medicine strategies for breast cancer based on a database on drug reactions". In: *Bioscience Reports* 39.9. ISSN: 0144-8463. DOI: 10.1042/BSR20190230. URL: https://doi.org/10.1042/BSR20190230 (visited on 07/27/2021).

Zhao, Huimin (July 15, 2022). "Synthetic Biology 2.0: The Dawn of a New Era". In: *ACS Synthetic Biology* 11.7, pp. 2221–2221. ISSN: 2161-5063, 2161-5063. DOI: 10.1021/acssynbio.2c00332. URL: https://pubs.acs.org/doi/10.1021/acssynbio.2c00332 (visited on 03/14/2023).

Zusammenfassung

Design und multikriterielle Optimierung von Zellklassifikationsschaltkreisen in der Tumortherapie

Maßgeschneiderte Frameworks, welche die Nutzung unvollständiger und verrauschter Daten ermöglichen, und damit die reale Umgebungen widerspiegeln, sind häufig entscheidend, um die Anforderungen eines bestimmten synthetischen Designproblems zu erfüllen. In dieser Arbeit konzentriere ich mich auf den Entwurf von synthetischen Schaltkreisen für die Diagnose und Behandlung von Krebs. Insbesondere entwickle ich Berechnungsrahmen für den logikbasierten Entwurf von Klassifikationschaltungen, wobei ich eine Reihe verschiedener Berechnungsparadigmen vom maschinellen Lernen bis zu evolutionären Algorithmen verwende. Zunächst konzentriere ich mich auf die Optimierung von Klassifikatoren für einzelne Schaltkreise entsprechend den Zielen und Beschränkungen, die durch den experimentellen Schaltkreisaufbau auferlegt werden. Ich nutze das Potenzial der logischen Programmierung, insbesondere der Antwortmengenprogrammierung, und schlage einen Arbeitsablauf für den Entwurf global optimaler logischer Klassifikatoren vor. Außerdem stelle ich einen alternativen, theoretischen Entwurf von Klassifikatoren vor, die aus mehreren Schaltkreisen bestehen, nämlich verteilte Klassifikatoren. Ich nutze die Vorteile von Ensembles, insbesondere die kollektive Entscheidungsfindung, um eine bessere Leistung bei heterogenen Daten zu erzielen. Um die Ensembles zu optimieren, entwickle ich einen benutzerdefinierten genetischen Algorithmus und überarbeite die Optimalitätskriterien für Klassifikatoren. Als Nächstes konzentriere ich mich auf die Verfeinerung der Bewertungsstrategien und die Erhöhung der Robustheit unserer Designs gegenüber neuen Daten. Schließlich untersuche ich alternative Anwendungen jenseits von Krebsklassifikatoren, um die Vielseitigkeit der vorgeschlagenen Methoden zu demonstrieren.

Eidesstattliche Erklärung

Name: Nowicka
Vorname: Melania Maria

Ich erkläre gegenüber der Freien Universität Berlin, dass ich die vorliegende Dissertation selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die vorliegende Arbeit ist frei von Plagiaten. Alle Ausführungen, die wörtlich oder inhaltlich aus anderen Schriften entnommen sind, habe ich als solche kenntlich gemacht. Diese Dissertation wurde in gleicher oder ähnlicher Form noch in keinem früheren Promotionsverfahren eingereicht.

Mit einer Prüfung meiner Arbeit durch ein Plagiatsprüfungsprogramm erkläre ich mich einverstanden.

Melania Maria Nowicka, Berlin, 29. Juni 2023