Toward Interpretable Machine Learning, with Applications to Large-scale Industrial Systems Data

Graziano Mita

This dissertation is submitted for the Degree of
Doctor of Philosophy
in the Doctoral School N. 130:
Computer Science, Telecommunications and Electronics of Paris
of the Sorbonne University

April 2021

Committee in charge:

Pietro Michiardi	EURECOM	Advisor
Serena Villata	Laboratoire I3S	Reviewer
Giovanni Neglia	INRIA	Reviewer
Zeynep Akata	University of Tubingen	Examiner
Maurizio Filippone	EURECOM	Examiner

To my beloved family.

Acknowledgments

I would like to thank my supervisor Prof. Pietro Michiardi, who pushed me to undertake with conviction this incredible journey. With his unconditional help and patience, Pietro has contributed to all my achievements, engaging in interesting discussions, sharing ideas, proposing solutions and new points of view. His support was crucial and sincere.

I would also like to extend my deepest appreciation to Prof. Maurizio Filippone and Prof. Paolo Papotti for their advice and time. Maurizio's unshakable optimism and Paolo's perseverance, together with their competence and patience, were precious elements along this path.

Special thanks to my colleagues at SAP Labs France, with whom I have created an emotional bond that goes beyond the simple working relationship. I cannot fail to mention the other professors, colleagues and friends from EURECOM, with whom I shared great times.

I have finally the opportunity to express eternal gratitude to my beloved family, my father Giuseppe, my mother Cosimina, my sister Francesca, and all my old friends, who have always been a constant presence in my life, despite the distance and hard times.

Abstract

Nowadays, Machine Learning and Artificial Intelligence (AI) are impacting the future of every industry and human being. The potential of such technologies is recognized in many application sectors. However, they meet the high barrier of *interpretability*, sometimes referred as *explainability*, that affects the so called *black-box* models. The lack of interpretability is a serious problem and prevents the usage of AI systems in fields involving critical decision-making, with significant consequences on human life.

In the literature, there are mainly two approaches to tackle machine learning interpretability: the former bypasses the issue by relying on naturally interpretable, *transparent methods*; the latter exploits *post-hoc interpretability* techniques, specifically designed to explain black-box methods. In both cases, interpretability depends on the comprehensibility of the data representation taken as input by the models.

The contributions presented in this work are two-fold. We first provide a general overview of explanations and interpretable machine learning, making connections with different fields, including sociology, psychology, and philosophy, introducing a taxonomy of popular explainability approaches and evaluation methods. We subsequently focus on *rule learning*, a specific family of transparent models, and propose a novel rule-based classification approach, based on monotone Boolean function synthesis: LIBRE. LIBRE is an ensemble method that combines the candidate rules learned by multiple bottom-up learners with a simple union, in order to obtain a final intepretable rule set. Our method overcomes most of the limitations of state-of-the-art competitors: it successfully deals with both balanced and imbalanced datasets, efficiently achieving superior performance and higher interpretability in real datasets.

Interpretability of data representations is central in our discussion on machine learning interpretability and constitutes the second broad contribution to this work. We restrict our attention to *disentangled representation learning*, and, in particular, VAE-based disentanglement methods to automatically learn representations consisting of semantically meaningful features. Disentangled representations are not only interpretable, but also useful to simplify

downstream tasks, for the benefit of the whole predictive chain. Recent contributions have demonstrated that disentanglement is impossible in purely unsupervised settings. Nevertheless, incorporating inductive biases on models and data may overcome such limitations. We present a new disentanglement method – IDVAE – with theoretical guarantees on disentanglement, deriving from the employment of an optimal exponential factorized prior, conditionally dependent on auxiliary variables complementing input observations. We additionally propose a semi-supervised version of our method. Our experimental campaign on well-established datasets in the literature shows that IDVAE often beats its competitors according to several disentanglement metrics.

Keywords: Interpretable Machine Learning, Explainable Artificial Intelligence, Rule Learning, Disentangled Representation Learning.

Preface

This thesis comprises the work on Interpretable Machine Learning carried out over the duration of my PhD.

Chapter 1 is a gentle introduction to the gigantic field of *interpretable machine learning*. It summarizes the exploration work of the first part of my research, with interesting references to different fields, including human sciences, well aligned with my classical background and strongly related to the concept of explanations and interpretability. This preliminary phase has been crucial for identifying the direction of the thesis, that can be mainly divided into two distinct parts, focusing on interpretability of machine learning models (*rule learning*) and data representations (*disentangled representation learning*), respectively.

Chapter 2 and **Chapter 3** extend the work presented in:

> Graziano Mita, Paolo Papotti, Maurizio Filippone, Pietro Michiardi. LIBRE*: Learning Interpretable Boolean Rule Ensembles*. Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics, AISTATS, 2020.

In particular, Chapter 2 submits a unified view of transparent, *rule learning* approaches, with a critical review of their main problems and possible solutions. Chapter 3 builds on top of the considerations of Chapter 2 and details our proposal to overcome such issues.

Chapter 4 and **Chapter 5**, expand on:

> Graziano Mita, Maurizio Filippone, Pietro Michiardi. *An Identifiable Double* VAE *For Disentangled Representations*. Under review by the 38th International Conference on Machine Learning, ICML, 2021.

Chapter 4 defines the problem of *learning disentangled representations*, reporting quantitative measures to evaluate disentanglement and practical examples where disentanglement is beneficial. Chapter 5 reviews the main VAE-based approaches from the literature to learn disentangled representation and analyzes the relation between model identifiability and disentanglement.

Moreover, it delineates our proposal to learn disentangled representation with theoretical guarantees.

Chapter 6 concludes this thesis with a summary of the main themes and contributions, followed by a brief discussion on future work.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER

ONE

INTERPRETABLE MACHINE LEARNING

Over the last decades, the terms *interpretability* and *explainability* have become popular in the machine learning research community. However, if we thought they were new concepts, we would be definitely wrong. Explanations have very ancient origins: philosophers have always tried to answer existential questions about why things happen, social scientists and psychologists have investigated the way humans communicate explanations to each other for years. Today, we are trying to extend those findings to mathematical models, in the hope of making them not only more intelligent and accurate, but also comprehensible to those who are not necessarily experts. This is particularly important when such models are used in critical application areas, where the decisions made by a machine may have severe consequences on human lives.

1.1 Overview

With the increasing availability of data and the recent technological advancements that allow people to access powerful hardware at accessible prices, machine learning has naturally become part of our everyday life. Nowadays, machine learning, and in particular deep learning models, are potentially adopted everywhere to solve practical problems. In some fields, such as computer vision, Aritificial Intelligence (AI) has already exceeded human capabilities, providing undeniable benefits. However, this huge diffusion has also highlighted important weaknesses: many AI applications, especially the ones that involve critical decision-making, see limited usage or are not appropriate, mainly due to ethical and legal problems, or the lack of trust from their users.

Motivated by the limitations above, the concept of interpretability of machine learning

models has seen a renewed interest in recent years. The basic idea is that by designing interpretable models, or by finding a way to explain opaque (black-box) models, the users can better understand their internal mechanisms and, as a consequence, trust these systems. There are, of course, other reasons for machine learning interpretability, including the need to comply with new laws and regulations.

Despite this intuitive definition, navigating through the immense literature about interpretable machine learning is quite complicated. Due to its intrinsic subjective nature that involves humans in the loop, the work on explainable AI covers different fields, not necessarily related to computer science or machine learning. If we want to design and implement intelligent systems that people can interpret, analyzing the process humans follow to explain decisions and evaluate explanations becomes essential. This takes us in the world of social sciences, that have studied human behavior for a long time, and represents the starting point to try to give a general and valid definition of interpretability in machine learning, identifying its properties and goals. Then, we propose a taxonomy to structure the work in the area. Chapter 1 is an indispensable step towards the next chapters: the time spent to explore and organize the literature helped us to mature a critical view of the field, moving us into specific research directions, leaving out others.

Outline of the chapter. In Section 1.2, we summarize the work carried out in social sciences about how and why humans explain things. This is preliminary to Section 1.3, where we focus on interpretability in the context of machine learning models, highlighting properties, desiderata and common types of explanations in the literature; we additionally report a general framework for evaluating explanations. In Section 1.4, we present a taxonomy of interpretable machine learning, mainly distinguishing between *transparent models* and *post-hoc explainability techniques*. We conclude the chapter with a discussion about what we think should be the future of interpretable machine learning; this section will also act as a bridge to the next chapters.

1.2 The Social Importance of Explaining Why

Despite its long history of research, that might be dated back to the invention of the first expert systems in the 1970s, there is not a formal and commonly accepted definition of machine learning interpretability, yet. A vast majority of recent work in the field assumes a personal view of what a good explanation is, neglecting many, if not all, the studies about human explanations. This might seem natural, considering the subjective nature of the concept of interpretability. Nevertheless ignoring the huge amount of work from sociology, psychology, and philosophy, about how humans define, communicate and evaluate explanations would be counterproductive.

Therefore, we start our journey by reviewing some insights from social sciences (Miller, 2019), so as to understand what people generally expect from explanations, and propose, in the next sections, a more rigorous definition of machine learning interpretability.

1.2.1 The Art of Explaining to Humans

In one of its philosophical papers, David K. Lewis claims that explaining an event means "*providing some information about its causal history. In an act of explaining, someone who is in possession of some information about the causal history of some event tries to convey it to someone else*" (Lewis, 1986). In other words, an explanation is both a *product* and a *process*: on the one hand it is a product because it can be seen as an answer to a why-question, given a presupposition such as "they did that. Why did they do it?", on the other hand it might also be considered as the cognitive process of deriving an explanation. Philosophers and psychologists agree on the fact that explanations refer to a cause (Salmon, 1989; Woodward, 2004) (in this work, we will not consider non-causal explanations, answers to questions like "what happened"). However, there are also other important properties to consider when discussing about explanations. Many explanations are *contrastive*: people do not usually ask why event P happened, but why event P happened instead of a different event Q. Answering these questions is generally easier than providing complete explanations. Explanations may also be *implicitly contrastive*, which makes it harder (or even infeasible) to figure out the event Q the user is implicitly referring to. Explanations are *selected*: people do not ask for the complete cause of an event, but select one or two causes to be the explanation. Finally, explanations may be *social*: in this case, they should be seen as a conversation with the goal of transferring knowledge, hence they should take into account how people interact.

Interpretable explanations. Throughout the whole thesis, the terms interpretability and explainability are used interchangeably and invoke a measure of how understandable an explanation is. More formally, in Biran and Cotton (2017) interpretability is defined as the degree to which an observer may understand why a given decision has been taken. This means that a certain concept, used to explain a given decision, may be transmitted in different forms and different degrees of complexities, depending on the listener.

The reasons behind an explanation. An obvious reason to ask for explanations is to understand why something happened. In this case, explanations are seen as a need for humans to find meaning. Deeper logics involving social interactions, like creating a shared meaning of something, transferring knowledge, influencing people's beliefs, and so on (Malle, 2004) are outside the scope of this work. In general, explanations might be required to persuade someone that a given decision is right (Lombrozo, 2006): then, providing the true reason might become of secondary importance.

The structure of an explanation. Different questions require different explanations: for example, asking why a certain event happened is not the same as asking where it happened. In this work, we focus on why questions only. One of the oldest and most known explanation models in philosophy is the Aristotle's *Four Causes* model. According to the Greek philosopher, we know something (and we can explain it) when we are able to identify its *causes*, that is when we know why something exists and why it exists as it is. This translates into four *causes* or *modes*: the former two, *material* and *formal*, refer to its composition; the latter two, *efficient* and *final*, regard their origin and goal. For instance, the material cause for a statue is the marble, the formal cause is the shape given to the marble, the essence cause is the sculptor's scalpel, and the final cause is the reason why the sculptor makes the statue. Each of these causes, either taken alone or all together, can be considered as reasonable explanations for why questions.

How people explain behavior. Research on *social attribution*, which studies how people explain behavior to others, constitutes the foundation for much of the work on explanations in general, and it is of great interest for AI explainability. Heider and Simmel (1944) are the first to demonstrate that humans tend to attribute *folk psychological concepts*, such as desire and intention, to objects. In one of their experiments, they asked the participants to watch a video with animated shapes moving around the screen, and to describe the scene. The participants described the movements of the objects as they were performed by humans. Then, Heider argued that the main difference between the human perception of objects and other humans is strongly related to the presence or lack of specific *intentions*. Many years later, in one of his most famous books (Malle, 2004), Malle proposes a conceptual framework for behavior explanations, formally distinguishing between *intentional* and *unintentional behaviors*: for unintentional behavior, people tend to offer only causes, while for intentional behavior people prefer more complex explanations, taking into account mental states, desires, background reasons and emotions. Also *norms* and *morals* have a huge impact on social attributions. When people explain immoral behaviors or behaviors that go against commonly accepted "unwritten rules", they are likely to include their personal thoughts and judgements in the explanations. In other words, explanations are biased by social beliefs. Unfortunately, human behavior, personal and social opinions are hard to model and encode into AI explanations.

1.2.2 Selecting and Evaluating Explanations

Explanations can be seen as a cognitive process that guides the generation and reception of explanations. Such process can be split into three steps (Miller, 2019): i) *causal connection*, where we identify the main causes of an event; ii) *explanation selection*, where a small subset of the causes is selected as the explanation, iii) *explanation evaluation*, usually performed by people to whom the explanation is addressed.

Causal connection. It is the process of identifying the causes of a fact, inferring them from observation and/or prior knowledge. It is obvious that people cannot simulate back all possible events to understand the associated causes. They use heuristics based on several criteria: people tend to focus more on abnormal/unusual causes (*abnormality of events*), intentional events receive more consideration than unintentional ones (*intention of events*), a major focus is attributed to recent and controllable events, not coincidences, to identify the causes of a fact (*timing and controllability of events*); changing the perspective, the causes associated to an event typically change (*perspective*). In Section 1.3.2, we will see that most machine learning models learn associations rather than causal relationships.

Explanation selection. Even when it is possible to establish all possible causes of a fact, it would be impossible for a human to understand them. Explanation selection is the process of selecting a subset of the causes identified in the previous step to provide an explanation. The work in this area shows that people usually select explanations according to criteria that are very similar to the ones used to identify them: explanations taking into account differences with respect to other events, abnormal conditions and intentional causes are more likely to be selected. In general, necessary causes are preferred to sufficient ones; goals are generally better explanations than preconditions, but preconditions and goals together are sometimes preferred.

Explanation evaluation. When individuals receive an explanation, they determine its quality according to several criteria. In his *Theory for Explanatory Coherence* (Thagard, 1989), Thagard argues that humans judge positively explanations that are aligned and coherent with their prior beliefs. Moreover, high quality explanations are simple (few causes) and general (they explain multiple events). This *simplicity principle* is followed by many interpretable machine learning models, including LIBRE (see Chapter 3 for more details). Surprisingly, while true and high probability causes are part of good explanations, they are not always related to explanations that people find useful (Hilton, 1996). Vasilyeva et al. (2015) also notice that explanations where the explanatory mode (material, formal, efficient, final) is well aligned with the goal of the question are preferred.

1.2.3 Communicating Explanations

According to the *conversational model of explanation* (Hilton, 1990), explanations can be seen as a two-stage process: i) the *diagnosis* of causality, where the explanation is actually "crafted" by identifying the main causes, and ii) the *conversation*, where the explanation is conveyed to someone. Because of this second step, explanations are subjected to the rules of conversations. Then, those explanations should contain causes that are *relevant* to the explainee, aligned both with his prior and shared knowledge between explainer and explainee.

The Grice's maxims (Greaves et al., 1999). They are a set of basic rules that should be followed to present an explanation. Although they are explicitly conceived for speeches, they naturally extend to any other conversation language. Grice identifies four classes of maxims that we summarize as follows. i) *Quality*: a) do not say things that you believe to be false, b) do not say things without sufficient evidence. ii) *Quantity*: a) make your contribution as informative as is required, b) do not make it more informative than is required. iii) *Relation*: a) Be relevant. iv) *Manner*: a) avoid obscurity of expression, b) avoid ambiguity, c) be concise and (d) be orderly. In a few words, an explanation should only contain necessary and relevant information.

Argumentation. A research study from Antaki and Leudar (1992) shows that a considerable amount of statements in explanations are *argumentative claim-backings*. When the explainer explicates or justifies something, he has to be ready to defend his claims. Interestingly, argumentation is dependent on what the explainee already knows, and focuses on abnormal factors as a way to empower the explanation. This confirms that good explanations must be relevant to both the question and the mental model of the explainee. Argumentative AI explanations are outside the scope of this thesis.

1.3 Interpretable Explanations in AI

When we talk about explainability and interpretability in machine learning, most of the concepts introduced in the previous section are valid and should rather be considered as necessary conditions to obtain "really interpretable" artificial intelligent systems. Unfortunately, today we are still far from this ideal goal: although many works, maybe unknowingly, tackle some of the desiderata for effective explanations, there is currently no work that satisfies all the characteristics covered in the previous section. Such limitations can be traced back to the lack of a commonly accepted definition of interpretable machine learning: indeed, popular definitions are either contrasting or incomplete, as they only consider a subset of goals; goals in isolation are not a sufficient condition to make a model interpretable.

1.3.1 Toward a General Definition of Interpretable Machine Learning

In the machine learning literature, there has been a proliferation of definitions about interpretable AI starting from 2016. It is not a coincidence that, in the same year, the European Parliament has adopted, for the first time, a set of regulations for the collection, storage and use of personal information, the *General Data Protection Regulation* (GDPR). In particular, Article 22 refers to *automated individual decision-making, including profiling*, and affirms a *right to explanation*, with a consequent strong impact on machine learning algorithms. With the GDPR, explainable AI becomes a real need also by law.

Limitations of popular definitions. Here below some of the most popular definitions about explainable machine learning:

> "*By explaining a prediction, we mean presenting textual or visual artifacts that provide qualitative understanding of the relationship between the instance's components and the model's prediction, [. . .] as a solution to the trusting a prediction problem.*" (Ribeiro et al., 2016).

> "*An interpretable explanation, or explanation, is a simple model, visualization, or text description that lies in an interpretable feature space and approximates a more complex model.*" (Herman, 2017).

> "*In the context of machine learning models, we define interpretability as the ability to explain or to present in understandable terms to a human.*" (Doshi-Velez and Kim, 2017).

> "*To intuitively understand a machine learning model, we need to visualize it, make it accessible to the senses.*" (Offert, 2017).

> "*Explainable Artificial Intelligence will create a suite of machine learning techniques that enables human users to understand, appropriately trust, and effectively manage the emerging generation of artificially intelligent partner.*" (Gunning, 2019).

These definitions have several drawbacks: i) they are sometimes too vague (Offert, 2017), ii) they are linked to a specific explanation type, like textual or visual, which are not the only alternatives, as we will see later; iii) they involve the concept of "understanding" which is completely dependent on the explainee; iv) they consider a subset of the desiderata of an explanation (we refer the reader to Section 1.3.2 for further details), such as informativeness and trust.

A general definition. With that in mind, Arrieta et al. (2020) have recently proposed a more general and coherent definition:

> "*Given an audience, an explainable Artificial Intelligence is one that produces details or reasons to make its functioning clear or easy to understand.*",

Here, the key is the presence of the three words "given an audience", that makes the concepts of clarity and simplicity specific for a given target explainee. Last but not least, this definition involves causal reasoning, that we identified as one of the main components of the explanation process, and "details" that refer to any other type of information we want to capture in the explanation.

1.3.2 Desiderata of Interpretable Explanations

Recent work from the literature on interpretable machine learning identify different desiderata for interpretable explanations. We have already seen that explanations can be considered as answers to "why-questions". If we restrict our focus on machine learning models, we might say that explanations answer the question "Why does the model give that output?". This question gives rise to a series of side-goals that, depending on the task and application area, might be the real reason why we need the explanation.

Informativeness. Every time we run a machine learning model to solve a given task, we obtain some information back, in the form of an output. However, by interpreting either the internal mechanisms of the model or the output (or both) we can obtain additional information, that can then be used by human decision makers to make better decisions. Interpretable machine learning aims at simplifying this process, making this "additional information" easier to understand, given the explainee and the solved task. All work carried out in this area has the informativeness goal in mind.

Causality. Overall, machine learning algorithms are not guaranteed to learn causal relationships; they rather learn associations among input data. Indeed, there could always exist unobserved variables that the machine learning model cannot take into account and that actually represent the true causes. Even when causal inference (Pearl, 2009) is possible, it usually relies on strong assumptions, that are not always satisfiable in practice. It is also true that causation involves correlation; thus, we may hope that by designing interpretable models or interpreting models, we could generate hypotheses that scientists could then test experimentally. While causality is one of the main reasons for explanations, surprisingly there are not many works on interpretable machine learning that explicitly set it as their goal.

Trustworthiness. Interpretability has often been studied in the light of trust (Ribeiro et al., 2016; Lipton, 2018). Trustworthiness is defined as the confidence that a model will act as expected. It is a property that every machine learning model should ideally respect, but trusting a model does not necessarily imply that a model is interpretable. Additionally, a perfect model, that always returns the exact output, is not automatically trustworthy; it might be the case for a model that makes "reasonable mistakes", for example in situations where also humans make mistakes.

Transferability. Transferability is another important goal for interpretable models (Ribeiro et al., 2016; Lipton, 2018). Being able to understand the decisions made by a model is a necessary condition to exploit the discovered knowledge in different contexts. In Section 2.4.2, we will see that this is true for rule-based systems, where

knowledge is encoded in the form of rules that can be continuously learned, removed, adapted to maintain their effectiveness even in presence of concept-drift (Lu et al., 2019). Disentanglement methods constitute another example of interpretable models, well suited for transfer learning tasks (see Section 4.4.2 for more details).

Fairness. Many works argue that interpretability is a practical way to increase fairness in machine learning models. We know that data is unfair, simply because society is unfair, too. The internal mechanisms of machine learning algorithms might amplify such problems. This is particularly relevant in sensible applications, where decisions made by a model impact human lives and cannot be based on ethical biases. The right to explanation, defined in the GDPR, regulates all cases where personal data are used: then, explanations must not only guarantee fairness, but also be provably correct and contestable. As will be discussed in Section 4.4.3, learning disentangled representations is a suitable way to achieve fair machine learning.

Interactivity. If we follow the view of explanations as conversations, some machine learning models (Langley et al., 2017) support interactions with the user who can question the model and obtain an answer. This implies the capability of the model to establish a dialogue and provide explanations.

Accessibility. Dealing with interpretable models also considerably affects their development process, from training to validation. Indeed, understanding how and why a given model returns a given output might allow even non-expert users to improve the model's performance, more easily than highly non-linear black-box models.

Other desiderata. Explanations and interpretable models have also been used to i) improve the **stability** of machine learning models (Lakkaraju et al., 2020), ii) increase the **robustness** against adversarial attacks (see Section 4.4.4), iii) preserve **privacy** when models process sensible information (Baron and Musolesi, 2020).

1.3.3 Properties of Interpretable Models

In the literature, there is a clear distinction between *interpretable-by-design* models, that are considered, to some degree, naturally interpretable, and *black-box* (or *opaque*) models, like neural networks, that, in contrast, require external techniques to interpret them, uncovering the meaning of their parameters. These two families of approaches are known as *transparency* and *post-hoc interpretability*, respectively.

This is just a first level of distinction: transparent models can be further classified according to three properties that reflect different levels of transparency: *simulatability*, *decompasability* and *algorithmic transparency*. Post-hoc interpretability techniques can

also be discerned according to i) the type of explanation (text, visual, etc...), ii) their dependence (or independence) from specific models, iii) their ability to provide either local or global explanations. We remind the reader that, according to the conversational model of explanation (Hilton, 1990), every time we generate an explanation we may communicate it in different ways and format, thus using different techniques, depending on the audience.

Transparency. Not all transparent models are the same: some of them are more interpretable than others. Thus, it is worth establishing three different levels of transparency: *simulatability*, *decompasability* and *algorithmic transparency*, where each level is also covered by the previous ones. For example, a simulatable model is decomposable and algorithmically transparent, but a decomposable model is not necessarily simulatable.

- **Simulatability.** A model is simulatable if a user can "contemplate the entire model at once" (Lipton, 2018). More precisely, given an input observation and the model's parameters, a user has to manually generate the output of the model, in reasonable time. This is possible only if the model is *simple* and *compact*. Simulatability suggests that no model is intrinsically interpretable: sparse linear models are more interpretable than the dense ones, but, for the same reason, a single perceptron neural network is more interpretable than a big decision tree.
- **Decomposability.** Even if users are not able to fully simulate the output of a model, they might still be able to follow and understand the output generation process. More formally, a model is decomposable if a user can intuitively explain each of its components: inputs, parameters and calculations. A simple and compact model that takes complex features as input is not decomposable.
- **Algorithmic transparency.** A model is algorithmically transparent if a user can totally explore it by means of mathematical analysis and methods. Linear models, independently from their size and sparsity, are algorithmically transparent as it is possible to reason about their loss shape. This is not the same for deep neural networks and their complex loss functions.

Post-hoc interpretability. Black-box models require external techniques to be interpreted. They can be classified according to three different levels:

- **Explanation type.** i) *Text explanations* provide explanations in textual format. This line of work also contains models that generate captions from images. ii) *Visual explanations* aim at explaining what a model has learned by means of visualizations. Some methods use dimensionality reduction techniques to represent input observations on two or three dimensions; methods from the computer vision community highlight the portion of the image/video the target model is focusing

on to generate its output. iii) *Explanations by example* rely on the assumption that a model should behave similarly when processing similar input observations. Then, the decision of a model is explained by means of one or more similar examples or proxies. iv) *Feature relevance* methods clarify the inner mechanisms of a model by computing a relevance score for each input feature: the higher the score, the higher the impact on the output.

- **Local vs global.** Local techniques consider a portion of the solution space, and provide explanations for a less complex subspace. Such techniques usually perturbate the input observation, for which we require an explanation, and approximate the target model with a simpler, interpretable one. Alternatively, it is possible to learn a transparent model that replicates the behavior of the entire target model.
- **Model-specific vs model-agnostic.** Some post-hoc interpretability techniques are constrained to work with a specific type or family of models, whereas others are model-independent (agnostic).

1.3.4 Evaluation Procedures

Doshi-Velez and Kim (2017) are the first to formally propose a taxonomy of evaluation approaches for interpretability, distinguishing among *application*, *human* and *function*-level evaluation.

Application level evaluation (real task). It is the more specific and costly evaluation procedure. It involves conducting human experiments within a real application: a transparent model, or a black-box model combined with a post-hoc model, runs in a real scenario and is evaluated by humans, better if domain experts. The quality of the explanation is measured according to its practical utility to solve the task at hand, in terms, for example, of better identification of errors, less discrimination, and so on.

Human level evaluation (simple task). It can be seen as a simplified application level evaluation, where experiments are carried out with a lay person instead of domain experts. This is particularly convenient because it does not require high-profile testers, hence it is easier to find more testers, but it is also limiting because it impacts the essence of real end-tasks. Potential experiments might be, for instance, *binary forced choices*, where the testers are presented with pairs of explanations and have to choose the best one, or *forward simulations*, where, given an input and the corresponding explanation, the testers have to simulate the output of the model.

Function level evaluation (proxy task). This evaluation procedure is the easiest to automatize as it does not involve humans in the loop. It relies on some proxies

to measure the interpretability of a model. Some models are better suited for this evaluation procedure than others. For example, a proxy for rule-based systems might be the number of rules, their size or a linear combination of them; a proxy for decision-trees might be their height, and so on, eventually penalizing models that lead to a significant performance drop. For other models, finding a good proxy is still an open problem.

1.4 A review of Explainability Approaches

On the basis of the taxonomy presented in the previous section, we will now review some of the most popular approaches for interpretable machine learning. A quick summary is illustrated in Figure 1.1.

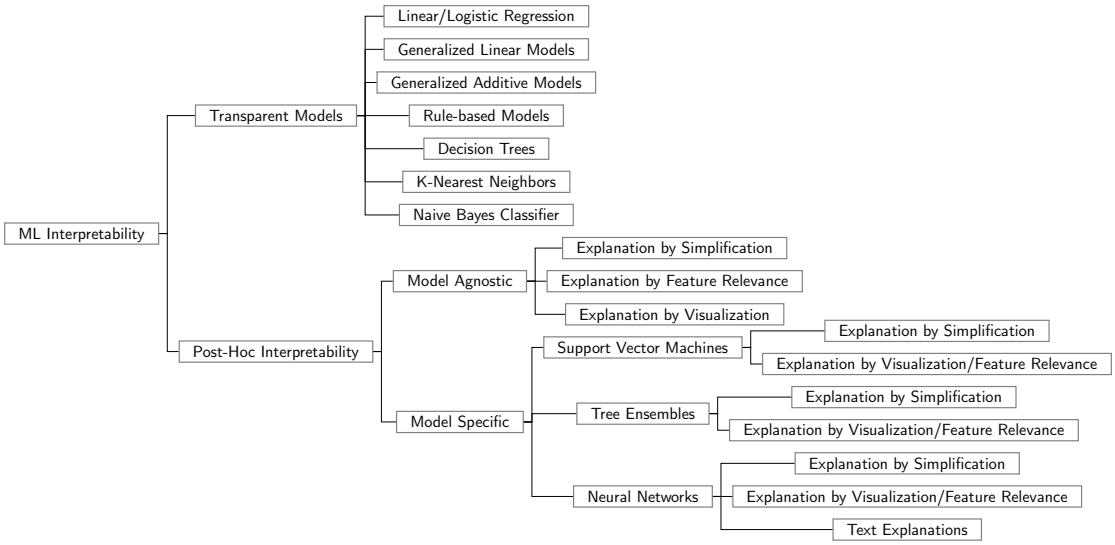


Figure 1.1: A taxonomy of machine learning interpretability approaches.

1.4.1 Transparent Models

One of the options to achieve machine learning interpretability is to use a subset of the models that are considered natively-interpretable, the so called transparent models. In this section, we are not going to present an extensive treatment of such methods, that are already well known; we will rather look to a representative subset of transparent models from an interpretability perspective, focusing on their transparency properties.

Linear/Logistic Regression. Linear regression (Bishop, 2006) predicts a continuous target as a linear combination of the input features. If we analyze the weights, we can have an idea about the feature importance. When the number of input features is high, *sparse linear models*, like LASSO (Tibshirani, 1996), are preferred because they push many weights to be close to zero, increasing the interpretability of the model. Logistic

regression (Bishop, 2006) is the equivalent of linear regression for (binary) classification tasks: rather than fitting a line (or hyperplane), logistic regression squeezes the output of the linear equation between zero and one, by applying the logistic function; the output can then be interpreted as the probability of a sample to belong to the positive class. Again, the weights associated to each feature are linked to their importance. The linear assumption is, at the same time, the main strength and limitation of such models: i) it is one of the reasons of success not only in statistics but also in other fields like medicine, social sciences, and so on; ii) assuming that the relation among features is linear is too restrictive for many scenarios (in this cases, non-linearity has to be hand-crafted, affecting the final interpretability). Despite their simplicity, it is not rare to apply post-hoc explainability (mainly visualization) techniques to explain linear models to non-experts.

Generalized Linear Models (GLM). In a linear regression model, the value of the target variable is assumed to follow a Gaussian distribution. Generalized Linear Models (McCullagh and Nelder, 1989) extend linear models by allowing outcome distributions from the exponential family. The expected value of such distributions is then computed as the result of a *link function* g, eventually non linear, applied to the weighed sum of features. Logistic regression is a special case of GLM, with a Bernoulli distribution and the logit function as link function. Depending on the choice of the outcome distribution and link function, we can get more or less transparent models, usually with better performance than a simple linear model.

Generalized Additive Models (GAM). Generalized Additive Models (Hastie and Tibshirani, 1990) can be seen as a further generalization of Generalized Linear Models, where the weighted sum of features is substituted by a sum of arbitrary functions, eventually nonlinear, of each feature. These functions can be either fixed or learned automatically, and this implies adding another stage in the interpretation of such models. Despite their performance (often worse than more complex models), they are considered interpretable and versatile enough to be used in many practical applications like finance, biology, healthcare.

Rule-based Models. Rule-based models are among the oldest and most studied models in machine learning (Fürnkranz et al., 2014). They use conditional rules to encode the functional relation between input data and the target concept. It is possible to learn either independent rules (rule sets) or mutually-exclusive rules (rule lists). Depending on the number and size of the rules, rule-based models can be simulatable, decomposable, or algorithmically transparent. Refer to Chapter 2 for more details.

Decision Trees. Decision Trees are used for both regression and classification problems. In the literature, there are many different decision tree algorithms like CART (Breiman et al., 1984), ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993). They are very popular in decision making contexts because of their intuitive hierarchical structure. Given an input, we can follow the path from the root to the leaf to obtain the output. The path can be easily mapped to conditional rules, whose form is easily understandable by humans. Depending on their size, the number of used features and the complexity of the features, they can belong to any of the three layered transparent levels. Alternatively, it is possible to compute feature importance scores. There is also a considerable amount of work in decision tree simplification (Quinlan, 1987b). Unfortunately, they do not always guarantee high performance, which is why we have tree ensembles like random forest (Breiman, 2001) that are, however, not directly interpretable.

K-Nearest Neighbors (KNN). K-Nearest Neighbors (Cover and Hart, 1967) is an instance-based classification method where input samples are classified by voting among the classes of its k nearest neighbors, according to a chosen distance measure. For regression problems, voting is an aggregate function, like mean or median. This approach resembles the way humans make similar decisions for similar events they experienced in the past. Again, KNN falls within the category of transparent models. Depending on the value of K and the choice of more or less interpretable distance measures, it can be located in one of the three transparent categories.

Naive Bayes Classifier. Naive Bayes Classifier is a probabilistic model based on the Bayes' theorem with a strong (naive) independence assumption among features. In summary, given an input $\mathbf{x} = (x_1, \ldots, x_n)$ with n features, and a class C_k, where k is the number of possible classes, it assigns to x the class probability $p(C_k|x_1, \ldots, x_n) \propto P(C_k) \prod_{i=1}^{n} p(x_i|C_k)$. The independence assumption makes this model interpretable as it is possible to evaluate the direct impact of each single feature on the target class. Depending on the number of features and their complexity, the Naive Bayes Classifier can satisfy simulatability, decomposability and algorithmic transparency.

1.4.2 Post-Hoc Interpretability

When a machine learning method is not transparent, an external method has to be designed and applied to the opaque model to explain its output. In Section 1.3.3, we have seen that such techniques can be either model agnostic or designed to explain specific models only. On the basis of the same taxonomy, we will briefly review some popular post-hoc interpretability techniques: among the model specific techniques, we will focus on those that have been studied the most in the literature, particularly the ones that explain support vector machines, (tree) ensembles and neural networks.

(a) Original image. **(b)** *Electric guitar.* **(c)** *Acoustic guitar.* **(d)** *Labrador.*

Figure 1.2: Example reported in Ribeiro et al. (2016). Google's Inception neural network generates the following predictions for the image in Figure 1.2a: *electic guitar* ($p = 0.32$), *acoustic guitar* ($p = 0.24$), *labrador* ($p = 0.21$). Figures 1.2b to 1.2d show the explanations provided by LIME for each prediction.

Model agnostic techniques. A considerable amount of work in interpretable machine learning has been carried out for *model agnostic explainability techniques*, designed to work with possibly any target model. The vast majority of these approaches aim at simplifying the target model, either globally or locally, to make its prediction procedure more transparent. Feature relevance and visualization approaches are popular, too.

- **Explanation by simplification.** The easiest way to explain a black-box model is to train a separate, more interpretable, explainer model that approximates its predictions, applying the so called *teacher-student* model (Tan et al., 2018), where the black-box has the role of a teacher and the interpretable model is a student that mimics the teacher's behavior. Decision trees and rule-based learners are popular student models but, in principle, any transparent model can be a suitable choice. When the student replicates the teacher locally, we have local explanations. Among the most known contributions to this approach we find LIME (Ribeiro et al., 2016), that supports many methods as local approximators. In Figure 1.2, we report an example where LIME uses super-pixels explanations to justify the output of a convolutional neural network. Alternatively, it is also possible to approximate the whole black-box model, finding a trade-off between predictive accuracy and simplicity. Similarly, a well-known approach in this case is G-REX (Konig et al., 2008), a rule-based genetic algorithm that extracts global rules from opaque models.
- **Explanation by feature relevance.** One of the possible ways to understand the decisions made by an algorithm is to measure the influence of each feature on its output. In a recent framework called SHAP (Shapley Additive Explanations) (Lundberg and Lee, 2017), a set of feature importance scores satisfying specific properties is computed for every prediction, showing better consistency with human intuition than previous approaches. The feature importance of individual prediction

has also been studied through the lenses of coalitional game theory (Strumbelj and Kononenko, 2010), local gradients (Robnik-Šikonja and Kononenko, 2008; Baehrens et al., 2010), sensitivity analysis (Cortez and Embrechts, 2011), influence functions (Koh and Liang, 2017) and saliency methods (Dabkowski and Gal, 2017).

- **Explanation by visualization.** Many model-agnostic visualization techniques are based on feature relevance and visualize the impact of single features on the output of a model. A Partial Dependency Plot (PDP) (Friedman, 2000) for a given feature shows how the average prediction of the target model varies as a function of that feature. The same goal is attained by Accumulated Local Effects plots (ALE) (Apley and Zhu, 2019). The equivalent to a PDP for individual instances is called Individual Conditional Expectation plot (ICE) (Goldstein et al., 2013). Other sets of visualization techniques can be found in Cortez and Embrechts (2011, 2013).

Post-hoc interpretability of support vector machines. In the simplest setting, support vector machines (SVM) (Cortes and Vapnik, 1995) learn a hyper-plane, in a high dimensional space, to separate instances belonging to two classes. The best hyperplane is the one allowing the largest separation, or *margin*. Despite their intuitive idea, support vector machines are considered, especially by non-experts, opaque models and require post-hoc techniques to be explained.

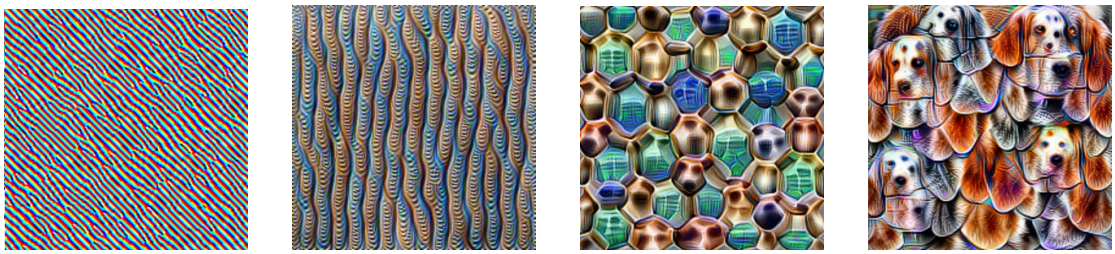
- **Explanation by simplification.** Rule-based learners appear to be the de-facto standard to simplify support vector machines. In Chaves et al. (2005); Barakat and Diederich (2006); Barakat and Bradley (2007), rules are extracted from support vectors; in Fu et al. (2004) hyperplane and support vectors are used together to induce a set of rules; in Nuñez et al. (2002, 2006) support and prototype vectors are combined to generate hyper-rectangles in the input space, then mapped to rules.
- **Explanation by visualization.** Visualization techniques for SVM are strictly linked to feature relevance and display the features that are most probably related to the predicted output. We find methods that can be applied for linear SVM (Rosenbaum et al., 2011), classification (Gaonkar et al., 2015) and regression (Üstün et al., 2007) tasks.

Post-hoc interpretability of tree ensembles. Tree ensembles are among the most used and accurate machine learning models, especially for structured data. In a tree ensemble, multiple decision trees are trained on a (random) subset of input features and combined by mean of an aggregation function. The resulting model is much more robust to noise than a single weak learner and achieves better generalization (Ho, 1998; Breiman, 2001). Unfortunately, this is also the main source of issues when it comes to explaining such models.

- **Explanation by simplification.** A considerable portion of work to explain tree ensembles follows the simplification approach, where a simpler model is trained to replicate the behavior of the ensemble. The authors of Domingos (1998) use C4.5 (Quinlan, 1993) to learn a less complex model from a set of samples labeled by the ensemble. Theoretically, their approach also works with other opaque models. In Deng (2019), a set of rules are extracted, cleaned and eventually refined from the ensemble, and are used to build a rule-based learner. In Hara and Hayashi (2018), a Bayesian model aims at learning the simplest, still accurate representation of a tree ensemble by simplifying its boundary.
- **Explanation by feature relevance.** Similarly to decision trees, it is possible to compute feature scores for tree ensembles as well. As far as we know, Breiman was the first to propose a feature importance measure for random forests (Breiman, 2001). In Auret and Aldrich (2012), significant features within a random forest are detected by including particular dummy variables in the system. In another work (Tolomei et al., 2017), the authors design an algorithm that outputs which feature should be changed and how to transform an instance into the opposite class for a binary random forest classifier.

Post-hoc interpretability of neural networks. Neural networks and, in general, deep learning methods have been successfully applied to many fields, often achieving or overcoming human accuracy levels for certain prediction tasks. Despite their incredible predictive power, neural networks struggle to be adopted in critical fields due to their intrinsic opaque nature. Recently, there has been a proliferation of publications aiming at improving the explainability of these black-box models, not only single-layer and multi-layer perceptrons (MLP), but also convolutional (CNN) and recurrent neural networks (RNN). Most of the proposed post-hoc interpretability techniques fall into the category of explanations by simplification, feature relevance and visualization.

- **Explanations by simplification.** Rule-based methods are the preferred transparent models when it comes to simplify neural networks. According to the authors, DEEPRED (Zilke et al., 2016) is the first rule extraction algorithm for deep neural networks: it extends an older decompositional approach designed for feedforward networks, CRED (Sato and Tsukimoto, 2001), by learning intermediate rules for every hidden layer. TREEVIEW (Thiagarajan et al., 2016) makes a hierarchical partitioning of the feature space learned by a fully connected deep model and build simpler, more interpretable meta-features, which reveals the iterative rejection of unlikely class labels until the correct association is predicted. Despite being a model-agnostic approach, LIME (Ribeiro et al., 2016) has been often used to interpret multiple types of neural networks by producing local explanations.



(a) Edges **(b)** Textures **(c)** Patterns **(d)** Objects

Figure 1.3: Feature visualization from Olah et al. (2017), learned by GoogLeNet (Szegedy et al., 2015), trained on the ImageNet dataset (Deng et al., 2009). Deeper layers (left-to-right) learn more complex features.

- **Explanation by feature relevance and visualization.** Feature relevance and the associated visualization methods represent a particularly important family of post-hoc explainability methods for neural networks. The authors of Montavon et al. (2017) adopt what they call *deep Taylor decomposition* to map the relevance of output neurons to the top layer that, in their case, consists of pixels. The final explanation is a heat-map of relevance scores associated to the input pixels that try to understand the regions of the input image the model focuses on to generate its output. In the same manner, DEEPLIFT (Shrikumar et al., 2017) computes the activation of each neuron and compares it to its *reference activation* to generate an importance score for that neuron. DEEPLIFT has considerable advantages over previously-proposed gradient-based methods. Zeiler et al. (2011) is another seminal work in this family that uses an adaptive deconvolutional network, where each layer learns a feature map that is projected back to the original image space. The strongest activations are then visualized with saliency maps and help to interpret what the model learned. The authors of (Mahendran and Vedaldi, 2015) propose to reconstruct input images from the CNN representations learned by intermediate layers, showing that they were able to capture accurate and human-interpretable information about the images, as shown in Figure 1.3. The work by Simonyan et al. (2014); Nguyen et al. (2016), instead, try to reconstruct the most representative image that maximizes a given class score, obtaining abstract images that resembles the true target classes. Interpretability of RNN has been less explored. We mainly find two approaches: i) understanding what a RNN has learned (Karpathy et al., 2015; Arras et al., 2017), ii) modifying the RNN architecture to make it more intelligible (Choi et al., 2016; Krakovna and Doshi-Velez, 2016).
- **Text explanations.** Some works propose to produce text explanations to clarify the output of neural networks, often CNN models. This methods usually rely on

RNN models to generate appropriate text descriptions for images (Hendricks et al., 2016; Donahue et al., 2017) or videos (Dong et al., 2017), eventually using attention mechanisms (Xiao et al., 2015; Xu et al., 2015).

1.5 Discussion

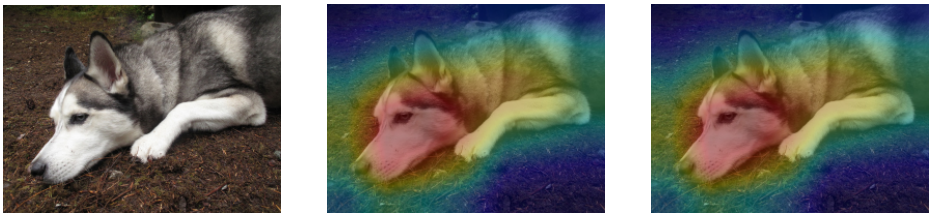
Getting oriented in the world of explainable AI is really tough. In the previous sections, we have seen that several fields, not necessarily related to computer science and machine learning, consider interpretability as a core concept for many, often critical, applications. Despite common belief, it is a very old research subject, that we tried to cover by providing a taxonomy and an overview of the most studied and popular approaches: transparent methods and post-hoc interpretability techniques.

At this point, it should be clear that every family and sub-family of explainable methods has its own advantages and disadvantages. We will now discuss more formally their criticalities, trying to figure out what might be the most desirable direction for future work. We want to specify that this is our personal view of the field, which pushed us to work in specific directions, disregarding others. Readers are invited to make up their own mind, that might eventually be completely opposite to ours.

1.5.1 On the Reliability of Post-Hoc Explanations

Every time we use post-hoc interpretability techniques to explain black-box models, there is the implicit belief that black-box models are better, in terms of performance, with respect to alternative transparent models. This is motivated by the wrong assumption that interpretability always comes at the expenses of predictive performance. It might actually be true for some fields, like computer vision, where opaque deep learning models are the de-facto standard, but this should not prevent us from designing increasingly accurate, still interpretable methods. In most practical scenarios, the space of solutions is so wide that it might indeed be possible to find simple models whose performance is equivalent to much complex ones.

In general, post-hoc interpretability techniques do not produce reliable explanations. If we focus on simplification methods for a moment, the goal is to train a transparent model that mimics the behavior of the black-box. Ideally, if the performance of the two is exactly the same, we would not need the black-box anymore. Unfortunately, even when performance is equal, we cannot be sure that the "reasoning" behind the decision made by the two models is the same. Indeed, the functions learned by the two models might generate the same output, still being different, that is using different features in different ways. On the other side, if the transparent model is only an approximation, there is no trivial way to trust its output as explanation. Simplification methods are not



(a) Image labeled as *siberian husky*.

(b) Explanation for *siberian husky*

(c) Explanation for *traverse flute*

Figure 1.4: Example reported in Rudin (2019). Saliency maps are not reliable explanation tools, because they provide similar explanations for different predictive outputs.

useless, but it might be misleading to consider them as completely truthful explanations.

Another interesting example of unreliability of post-hoc explanations is represented by all the feature relevance techniques that explain the output of opaque models, especially neural networks that process images, through saliency maps, heat-maps or other related visualization methods. The intuition is that, by figuring out which are the portions of the input where the network is focusing to generate its output, we can understand something about its decision process. Unfortunately, these techniques are able to reasonably recognize what part of the input is neglected by the model, but they do not provide any information about how the remaining information is used. Rudin (2019) showed that when we use saliency maps to explain, for example, CNN classifiers, they give similar, if not identical, explanations for each class, including wrong classes, simply because they focus on edges and boundaries, as shown in Figure 1.4.

We might continue giving other examples; however, the main idea is that every time an external model is used to explain a target model, we should expect to face one of the problems described above.

1.5.2 Challenges and Opportunities

In the light of the discussion above, a natural way of avoiding the issues associated to post-hoc explanations is to design directly transparent models, or models that satisfy specific interpretable properties by design, where explanations reflect the true decision process. This leads to other challenges, not necessarily easier to solve, and new opportunities.

Key issues of transparent models. Post-hoc methods are usually based on derivatives and can be optimized by gradient-descent. Interpretability constraints of transparent methods often lead, by contrast, to combinatorial optimization problems, known to be computationally hard to solve. Moreover, designing transparent methods is just a link in the chain: if we aim at obtaining performance that is in line with black-box models,

we need to spend much more time on the data knowledge discovery process, selecting and designing meaningful and high predictive features. The results are worth the effort since we can then rely on models and data that we fully understand. There is also another underestimated issue: in a world where researchers, data scientists and developers are trained to deploy deep learning models, expertise in interpretable methods is rare. Unfortunately, there are cases where post-hoc techniques are the only available option: some companies might have proprietary opaque models for which they do not want to reveal internal details, or that are being used without particular issues for a long period and there is resilience in substituting them.

Transparent models are the way. The preliminary analysis and exploration phase on explainable AI in this introductory chapter allowed us to get a general understanding of the field and the possible research directions. It has been definitely time-consuming, considering the high number of topics it covers, but essential to decide the next steps in our research. At this point, our opinion and preferences on explainable AI should be clear. If interpretability is a major constraint, we believe that transparent models are the way to go. Even when transparent models are wrong, explanations are still meaningful and allow us to understand what leads to the error, and improve the model. Motivated by this belief, we have explored two of the possibly infinite paths of explainable AI, that we are going to detail in the next chapters: i) *rule learning*, that is historically the oldest field related to interpretable machine learning and ii) *disentangled representation learning*, that aims at making modern deep learning generative methods interpretable.

CHAPTER

TWO

RULE LEARNING

Rule Learning is probably the oldest and most investigated field in machine learning and data mining. It plays an important role in the knowledge data discovery process by extracting useful information from data in the form of rules: the human-readable structure of rules makes them understandable by the users who can have an immediate feedback from the output of the machine learning model. Even when the learned rules are not always correct, they still reflect what the model actually computes and help to simplify the decision making process. Although it might seem a simple procedure, learning rules from data is a computationally expensive process, involving many building blocks, that can be implemented in different ways, each with their own advantages and disadvantages. Their interconnection determines the properties of the resulting rule-based methods, that we carefully organize into a general rule learning framework.

2.1 Overview

The first references to rules in the literature date back at least to 1970s, with the rise of the first expert systems (Jackson, 1998; Leondes, 2002). Expert systems were employed as diagnostic tools especially in medicine and biology, emulating the decision-making ability of a human expert through IF-THEN-ELSE rules. Since then, research on rule-based systems has never stopped and rule learning has become, without a doubt, one of the most studied and well defined field in the data mining and machine learning community.

Rule learning can be broken in two main categories: *descriptive rule discovery* and *predictive rule learning*. The former aims at finding out patterns and regularities in the data for exploration purposes, while the latter focuses on discovering rules that

identify particular targets or concepts of interest. Descriptive rule learning does not focus on predictive performance, but on the statistical significance of the discovered rules. The two main tasks for descriptive rule learning are *subgroup discovery* (Klösgen, 1996; Wrobel, 1997) where, as the name suggests, the purpose is to find subgroups according to a property of interest, and *association rule discovery* (Agrawal and Srikant, 1994; Pasquier et al., 1999; Borgelt, 2005), that extracts useful co-occurrences in the data.

In this work, we focus on predictive rule learning, that we investigate in the wider context of interpretable transparent models. Indeed, as anticipated in the previous chapter, the concepts learned by rule-based systems come in the form of rules that can be easily understood by humans, making them a suitable choice for critical applications.

Outline of the chapter. In Section 2.2, we formally define the main assumptions and concepts used to describe the rule learning task: we fix some notational conventions in terms of input data and rule representation. In Section 2.3, we present a unified view of the rule learning process that consists of three main steps: i) *feature construction*, ii) *rule induction*, and iii) *hypothesis construction*. The rule construction step necessarily implies *rule evaluation*, briefly reviewed in Section 2.3.3. In Section 2.3.5, we also report some popular techniques applied to avoid *overfitting* and obtain more general rules. Section 2.4 concludes the chapter with a concise but complete and critic view of the most popular and impacting rule-based methods. This analysis allows us to bring to light some of the most recurring problems in the field and acts as a bridge to the next chapter, where we actually propose our solution to these issues.

2.2 Problem Definition

Predictive rule learning can be defined, informally, as the task of learning rules from a training dataset with the goal of classifying new samples according to a given target. Input samples in the training dataset consist of a set of feature values and a class label. In the easiest settings, when a sample satisfies all the conditions of a rule, it is classified according to the class of that rule. Although the definition above is intuitive, it neglects important constraints for the learning task that are generally known as *language bias* of the learning problem: i) the *data description language* defining the input data format, and ii) the *hypothesis description language* describing the structure of the rules. Finally, a *coverage function* establishes the relation between data and hypothesis description language, allowing both to learn the relation between the features and the target, and to apply it to new data.

2.2.1 Data Description Language

The input data for the rule-based classifier consists of a set of records described by a set of features $\mathcal{F} = \{f_1, \cdots, f_F\}$ and a classification label $c \in \mathcal{C}$. Then, each record $\mathbf{x}_i$ can be written in *feature-value format* as: $\mathbf{x}_i = (v_{1i}, \ldots, v_{Fi}, c_i)$, where v_{ji} is the value that feature f_j takes for the input record x_i, and c_i is the corresponding classification label. Input features can be either categorical or numerical. In order not to make things unnecessarily harder, we assume $c \in \{0, 1\}$. In Section 2.3.1, we will observe that the original features $\mathcal{F}$ are transformed, either implicitly or explicitly, into a binary set of features $\mathcal{B}$: such binary features will be the building blocks of the learned rules.

2.2.2 Hypothesis Description Language

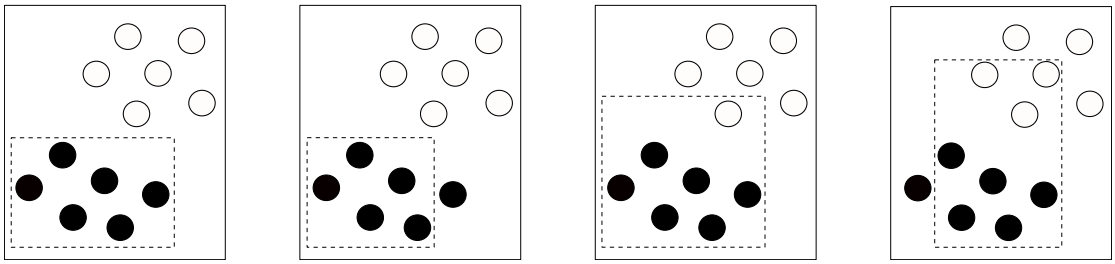
A rule learning model learns rules from feature-value datasets with the following form:

$$\underbrace{\text{IF condition}_1 \text{ AND} \ldots \text{AND condition}_L}_{body} \; \underbrace{\text{THEN target label}}_{head} \tag{2.1}$$

The condition part of the rule is called *antecedent* (or *body*), whereas the final part is called *consequent* or *head*. The antecedent consists of one or more Boolean *conditions* (or *atoms*) connected through the Boolean AND operator. Each condition corresponds to a check on a particular property of the target class, expressed as a function of the input features: in the simplest formulation, rule conditions have the form f_i OP v, where v is a possible value for feature f_i, and OP is a comparison operator. For discrete features, OP corresponds to the equality operator, while for numerical features it can be any operator among the following ones: $=, \neq, <, >, \leq, \geq$. Any ordinal relation can also be expressed in range format. One or more rules makes an *hypothesis* (or *theory*): in a *rule set*, all rules are equally important and are combined through the OR operator; in a *rule list* (or *decision list*), on the contrary, rules are mutually-exclusive and can be seen as a list of IF-ELSEIF statements. If not specified, we refer to rule sets.

2.2.3 Coverage Function

A rule r *covers* a given sample if all its conditions are true for that sample. We indicate coverage with an upper hat and non-coverage with an upper bar. More precisely, if we denote by $\mathcal{P}$ a set of positive samples and $\mathcal{N}$ a set of negative samples, then $\hat{\mathcal{P}}(r)$ and $\hat{\mathcal{N}}(r)$ represent the set of positive and negative samples covered by rule r, respectively; $\hat{P}(r) = |\hat{\mathcal{P}}(r)|$ and $\hat{N}(r) = |\hat{\mathcal{N}}(r)|$ represent their size. Equivalently, $\bar{\mathcal{P}}(r)$ and $\bar{\mathcal{N}}(r)$ denote the set of positive and negative samples not covered by rule r, respectively; $\bar{P}(r) = |\bar{\mathcal{P}}(r)|$ and $\bar{N}(r) = \bar{\mathcal{N}}(r)$ indicate their size. Given the total number of positive and negative samples, $P = |\mathcal{P}|$ and $N = |\mathcal{N}|$, we also define: $\hat{\pi} = \frac{\hat{P}}{P}$ (true positive rate), $\hat{\nu} = \frac{\hat{N}}{N}$ (false positive rate), $\bar{\pi} = \frac{\bar{P}}{P}$ (false negative rate), $\bar{\nu} = \frac{\bar{N}}{N}$ (true negative rate).



(a) complete and consistent

(b) incomplete and consistent

(c) complete and inconsistent

(d) incomplete and inconsistent

Figure 2.1: Completeness and consistency of rule-set $\mathcal{R}$. Target concept in black.

2.2.4 Predictive Rule Learning

At this point, we can finally give a more formal definition of predictive rule learning.

Definition 2.2.1. Given a data description language, a hypothesis description language, a target class, a coverage function and a set of positive and negative samples according to the target class, predictive rule learning aims at finding a hypothesis (or theory) $\mathcal{R}$ in the hypothesis description language that describes the target class.

Completeness and Consistency. The target class is also known as *target concept*. In ideal situations, without noise and errors, where positive and negative samples do not overlap, it is possible to learn a hypothesis, given a proper hypothesis description language, that is *complete* and *consistent* with the target concept (refer to Figure 2.1).

Definition 2.2.2. A hypothesis describing a target concept is: *complete* if it assigns the target class label to all the samples belonging to the target concept; *consistent* if it does not assign the target class label to samples that do not belong to the target class.

It is clear that, in realistic scenarios with noisy data and overlapping concepts, completeness and consistency are not only impossible to obtain, but also undesirable, as they would easily lead to overfitting. Even in the absence of noise, with perfectly separable samples according to the target concept, the chosen hypothesis description language might not allow to learn a complete and consistent set of rules. In practice, completeness and consistency are often relaxed and replaced by weaker conditions like partial coverage of the target samples, predictive accuracy, or other metrics.

Dealing with multiple classes. In the definitions above, we implicitly assumed to deal with *binary classification problems*, where the positive class is the target concept. Therefore, learning a hypothesis for the target class is sufficient to fully characterize the non-target class, too. When we deal with more than two classes/concepts, it is possible

to iteratively apply single concept learning to the different concepts, transforming the multiclass learning problem into *multiple concept learning tasks*. In this case, we might also need a strategy to manage conflicts, that is to decide the output class when different rules return different outputs.

2.3 Rule Learning Process

We present the rule learning process with a unified framework that allows us to analyze and compare most of the known rule-based methods. We split the rule learning process into three distinct steps:

- **Feature Construction.** The first step of any rule learning model is to transform the set of input *attribute-value features* in *binary features*. Some algorithms do this step implicitly, integrating it in the rule induction process; others have a separated feature construction operation before learning the rules.
- **Rule Construction.** This step, also known as *rule induction* or *rule learning*, establishes how single rules are learned. Each rule is typically induced for a specific target class or concept. Most of the rule learning methods learn the set of conditions by using heuristics, but smarter and more complex approaches are possible, of course. Rule construction inevitably involves **rule evaluation**.
- **Hypothesis Construction.** Multiple rules are usually needed to define a hypothesis. We can mainly choose to learn *unordered rule sets* or *ordered rule lists* (*decision lists*), each with their advantages and disadvantages.

In addition, many rule-based methods apply **pruning techniques** to fight overfitting and increase the generalization capabilities of the learned hypothesis. This can be considered either as a separate step after the hypothesis construction, or incorporated within the rule construction process. Predictive performance might be further improved by combining weak hypothesis with rule ensembles, often – but not always – sacrificing the interpretability of the final model.

2.3.1 Feature Construction

Any rule learning model performs, either explicitly (Lavrač et al., 1991; Lavrac and Dzeroski, 1993) or implicitly (Clark et al., 1987; Cohen, 1995), a feature transformation operation to convert the original input features into binary features that are then used to build the antecedent of the rules. Binary features can be distinguished between *propositional* and *relational* features.

- **Propositional features.** When a binary feature refers to a single input feature, we call it propositional feature. A propositional feature is usually in the form

Samples		Binary Features				
ID	LABEL	b_1	b_2	b_3	...	b_B
1	True	False	True	False	...	True
2	True	True	False	False	...	False
3	False	False	False	False	...	True

Figure 2.2: Example of covering table with three input samples and B binary features.

f_i OP v, where $f_i \in \mathcal{F}$, OP is a comparison operator, and v is a value in the domain of the input feature f_i.

- **Relational features.** When a binary feature refers to multiple input features, we call it relational feature. We can have a simple comparison between two input features like f_i OP f_j, or more complex conditions like $g(f_i, \ldots, f_j)$, where $g(\cdot)$ can be any operation or transformation that returns true or false. In our work, we do not deal with relational features, but they can always be hand-crafted and included as additional binary features before rule induction, if feature construction and rule induction happen as separate steps.

From input to binary features. Constructing a good set of binary features is not an easy task: it is a domain dependent operation and might become computationally expensive, especially when most of the original features are numerical. We rely on *covering tables* to build binary features. A covering table (Figure 2.2) is a binary table where the original records are represented in terms of binary features.

In Algorithm 1, we report a simple algorithm (Gamberger and Lavrac, 2002) to build propositional features for binary classification tasks. It can be shown that this algorithm generates a *sufficiently reach* binary feature set that allows to build a complete and consistent set of rules, if feasible (Fürnkranz et al., 2014, Section 4.4). It receives as input the set of positive and negative training samples $(\mathcal{D}_+, \mathcal{D}_-)$ and the set of original features $\mathcal{F}$. Then, if a feature f is discrete, it generates binary features $f = v$ for all the values appearing in the positive samples $(v \in \mathcal{V}_+)$, and $f \neq v$ for all the values in the negative samples $(v \in \mathcal{V}_-)$. If a feature f is numerical, it first sorts all possible values of f, identifies couples of consecutive values v_i, v_{i+1} belonging to different classes, computes the mean $v = (v_i + v_{i+1})/2$ and then generates binary features as follows: if the smaller value v_i belongs to the positive class (and the greater value v_{i+1} to the negative class) the binary feature $f < v$ is built; $f \geq v$ otherwise.

Algorithm 1 GENERATEBINARYFEATURES $(\mathcal{D}_+, \mathcal{D}_-, \mathcal{F})$

$\mathcal{B} = \emptyset$
for $f \in \mathcal{F}$ **do**
  $\mathcal{V}_+$ = set of values of f appearing in the samples in $\mathcal{D}_+$
  $\mathcal{V}_-$ = set of values of f appearing in the samples in $\mathcal{D}_-$
  if A *is discrete* **then**
    for $v \in \mathcal{V}_+$ **do** $\mathcal{B} = \mathcal{B} \cup \{f = v\}$
    for $v \in \mathcal{V}_-$ **do** $\mathcal{B} = \mathcal{B} \cup \{f \neq v\}$
  end
  if A *is numerical* **then**
    $\mathcal{V} = sort(\mathcal{V}_+ \cup \mathcal{V}_-)$
    for $i \in \{1, \cdots, |\mathcal{V}| - 1\}$ **do**
      $v = (v_i + v_{i+1})/2$
      if $v_i \in \mathcal{V}_+$ AND $v_{i+1} \in \mathcal{V}_-$ **then** $\mathcal{B} = \mathcal{B} \cup \{f < v\}$
      if $v_i \in \mathcal{V}_-$ AND $v_{i+1} \in \mathcal{V}_+$ **then** $\mathcal{B} = \mathcal{B} \cup \{f \geq v\}$
    end
  end
end
return $\mathcal{B}$

Missing values. By definition, covering tables contain true and false values only. However, the original features, especially in real datasets, may contain missing values too. There are several strategies to deal with missing values. Here, we report the three main families: i) **delete strategy**: it removes all records containing at least one missing value with a consequent reduction of the training data; **default value strategy**: it replaces missing values with a default value that can be True, False or a special value; iii) **guess value strategy**: it replaces missing values either with the most common value (usually the mean or mode) or with the value predicted by a classifier separately trained for each feature (more complex strategies can eventually output a distribution of values). These approaches have all their advantages and disadvantages, whose discussion is out of the scope of this work. Multiple strategies can eventually be used together.

2.3.2 Rule Construction

The rule learning process is an optimization problem where the goal is to find the best rule(s) according to a given quality measure, by traversing a search space, also known as hypothesis space, made of all the possible rules (Mitchell, 1982).

In Algorithm 2, we summarize the main steps of a general algorithm to learn the best rule from a set of positive and negative training samples $(\mathcal{D}_+, \mathcal{D}_-)$, characterized by a set of binary features $\mathcal{B}$, according to a given quality measure computed within EVALUATERULE. LEARNBESTRULE starts with one or more candidate rules that are

Algorithm 2 LearnBestRule $(\mathcal{D}_+, \mathcal{D}_-, \mathcal{B})$

```
r_best = InitializeRule(D+, D−)
h_best = EvaluateRule(r_best)
R = {r_best}
while R ≠ ∅ do
    R_cand = SelectCandidateRules(R, D+, D−)
    R = R \ R_cand
    for r ∈ R_cand do
        ρ(r) ∈ RefineRule(r, D+, D−)
        for r' ∈ ρ(r) do
            if RuleStopCondition (r', D+, D−) then
                continue
            end
            h' = EvaluateRule(r')
            R = InsertSort(r', R)
            if h'>h' then
                h_best = h'
                r_best = r'
            end
        end
    end
    R = FilterRules(R, D+, D−)
end
return r_best
```

stored in $\mathcal{R}$, in ascending order according to the chosen quality metrics. At each iteration, a subset of the candidate rules is selected and slightly modified with a RefineRule operator that can, for example, add or remove conditions, or perform random operations. Every time a rule is changed, it is evaluated and inserted into the sorted set $\mathcal{R}$. Most of the algorithms also use a stop condition preventing the rule to be further modified. Between consecutive operations, we can eventually apply FilterRules to reduce the size of $\mathcal{R}$. At the end of the process, the best rule r_{best} is returned.

The rule induction step of many rule-based models proposed in the literature can be described with Algorithm 2: they mainly differ in the implementation of the internal functions (in upper case). InitializeRule and RefineRule define the *search strategy*, SelectCandidateRules and FilterRules the *search algorithm*, and EvaluateRule characterizes the *search heuristic*; RuleStopCondition is usually used to avoid overfitting. We will analyze each of the following sub-modules in the next paragraphs.

Search Strategies. A search strategy defines how the search space is traversed to learn a new rule. Intuitively, we might start, for instance, with an empty rule, considered

as the most general rule, and add conditions to make it more specific, such that it can cover as many positive samples and as few negative samples as possible. We might also follow the opposite strategy, that is to say to start from a very specific rule and make it more general. A more formal definition of these concepts is given below:

Definition 2.3.1. Let r' and r'' be two rules. If r' covers all the samples covered by r'', we say that r' is more general than r'' and r'' is more specific than r'.

We navigate the search space by applying the so called *refinement operators*, already encountered in the previous section. We can therefore implement different search strategies by using suitable rule initializers and refinement operators. Each of the following strategies has its own advantages and disadvantages that will be discussed in more detail in Section 2.4.1.

- **Top-down strategy.** Also known as *general-to-specific* strategy. InitializeRule returns one or more empty rules, whereas RefineRule implements a specialization operator. In the easiest case, the specialization operator adds a new condition to the antecedent, but more complex operations are possible. AQ (Michalski et al., 1986), PRISM (Cendrowska, 1987), CN2 (Clark and Niblett, 1989), FOIL (Quinlan and Cameron-Jones, 1993), and RIPPER (Cohen, 1995) are examples of widely used top-down learners.

- **Bottom-up strategy.** Also known as *specific-to-general* strategy. InitializeRule returns one or more randomly selected samples in the covering table (also called *seed samples*) belonging to the target label, whereas RefineRule implements a generalization operator. In the easiest case, the generalization operator removes a condition from the antecedent, but, again, more complex operations are possible. Popular bottom-up learners are GOLEM (Muggleton and Feng, 1990) and MODLEM (Grzymala-busse and Stefanowski, 2001).

- **Mixed strategy.** Also known as *bidirectional search*. InitializeRule can return an empty rule, a seed sample, or a randomly generated rule. In other words, the rule induction process can start from any point of the search space. RefineRule can perform either a specialization or a generalization operation, or eventually both. Popular representatives of this strategy are JOJO (Fensel and Wiese, 1993) and ATRIS (Mladenic, 1993)

Search Algorithms. A naive algorithm to implement LearnBestRule enumerates all possible rules, evaluates them, and returns the best solution according to a given evaluation measure. Although the ordering of the search space may allow to reduce the number of candidate rules and, consequently, the research time, *exhaustive search algorithms* are very inefficient: they might still be used when constrained to output rules

with a small maximum rule length (Rivest, 1987). That is why the vast majority of rule learning methods use *heuristics*, eventually with a *randomness* component, to prune the search space, aiming to find a trade-off between performance and simplicity, instead of learning the best rule ever. Several works (Webb, 1993; Quinlan and Cameron-Jones, 1995; Janssen and Fürnkranz, 2009) verified that heuristic search, especially in datasets with noise, is less prone to overfitting and usually better than exhaustive search that suffers from the so called *over-searching* problem.

- **Exhaustive search.** Exhaustive search algorithms explore the whole search space to discover the best rule. In practice, they look for rules up to a given length and often employ stop conditions to prune the search space. There are three main approaches: *best-first*, *ordered*, and *level-wise*. Best-first approaches like PROGOL (Muggleton, 1995) select the best candidate rule and try all possible refinements, without applying any FILTERRULES operator. Ordered approaches like OPUS (Webb, 1995) assume a natural or artificial order in the search space (features are sorted according to their ability to discriminate the target concept) and traverse it by being sure to visit every rule once. Level-wise approaches like APRIORI (Agrawal and Srikant, 1994) generate all possible rules (feature sets) with a given feature length before considering the following length.
- **Heuristic search.** *Hill-climbing* is a popular method in this family: in Algorithm 2, only the output of the best refinement operator proceeds to the next iteration. Although hill-climbing is highly efficient, it suffers from myopia. To overcome this problem, multiple refinement steps might be applied before selecting the best rule, like done in ATRIS (Mladenic, 1993). Other methods like AQ (Michalski et al., 1986) and CN2 (Clark and Niblett, 1989) follow a *beam search* strategy, considering a *beam* of candidate solutions, instead of a single candidate, at each iteration.
- **Stochastic search.** In stochastic search algorithms, REFINERULE is allowed to perform random operations that move the learning process in a completely different area of the search space. In this category, we find simulated annealing (Kirkpatrick et al., 1983) and genetic algorithms-based (Goldberg, 1989) approaches.

2.3.3 Rule Evaluation

Rule evaluation metrics measure the quality of a rule and guide the rule learning process by selecting or discarding candidate rules. Intuitively, a good evaluation measure should prefer rules with high number of positive covered samples $\hat{P}$ (completeness) and negative uncovered samples $\bar{N}$ (consistency), as defined in Section 2.2. This is equivalent to minimizing $\hat{N}$. Most widely used evaluation measures define a linear (or nonlinear) combination of $\hat{P}$ and $\hat{N}$.

Measure	Formula
PositiveCoverage	$= \hat{P}$
NegativeCoverage	$= \bar{N} = N - \hat{N}$
Sensitivity (Recall, TruePositiveRate)	$= \frac{\hat{P}}{P} = \hat{\pi}$
Specificity (TrueNegativeRate)	$= \frac{\bar{N}}{N} = \bar{\nu}$
Support	$= \frac{\hat{P}}{P+N}$
Coverage	$= \frac{\hat{P}+\hat{N}}{P+N}$
CoverageDifference	$= \hat{P} - \hat{N}$
Accuracy	$= \frac{\hat{P}+\bar{N}}{P+N}$
RateDifference	$= \hat{\pi} - \hat{\nu}$
LinearCost	$= a \cdot \hat{P} - b \cdot \hat{N}$
RelativeLinearCost	$= a \cdot \frac{\hat{P}}{P} - b \cdot \frac{\hat{N}}{N} = a \cdot \hat{\pi} - b \cdot \hat{\nu}$
Precision (Confidence)	$= \frac{\hat{P}}{\hat{P}+\hat{N}}$
Exclusiveness	$= \frac{\hat{\pi}}{\hat{\pi}+\bar{\nu}}$
InformationContent	$= -\log \frac{\hat{P}}{\hat{P}+\hat{N}}$
Entropy	$-(\frac{\hat{P}}{\hat{P}+\hat{N}} \cdot \log \frac{\hat{P}}{\hat{P}+\hat{N}} + \frac{\hat{N}}{\hat{P}+\hat{N}} \cdot \log \frac{\hat{N}}{\hat{P}+\hat{N}})$
FMeasure	$= \frac{(\beta^2+1) \cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$

Table 2.1: Popular predictive performance rule evaluation metrics.

Predictive performance metrics. Given a dataset with P positive and N negative samples, predictive performance metrics are defined in terms of $\hat{P}$ and $\bar{N}$. The goal is to maximize $\hat{P}$ and $\bar{N}$. In Table 2.1, we report some of the most known and used metrics. A full review can be found in Fürnkranz et al. (2014). We can distinguish among:

- **Elementary metrics.** They are defined in terms of either $\hat{P}$ or $\hat{N}$ (or equivalently $\bar{N}$). In this list, we find *positive coverage*, *negative coverage*, their corresponding ratios – *sensitivity* and *specificity*, also known in association rule discovery as positive and negative *local support* respectively – and *support*. Support and sensitivity are equivalent, they only differ in the normalization constant.
- **Linear metrics.** They are linear combinations of elementary metrics. Examples of linear metrics are: *coverage*, *coverage difference*, *accuracy*, *rate difference*; *linear cost* and *relative linear cost* generalize *coverage difference* and *rate difference*.

Theory for positive class	Theory for negative class
IF $condition_{11}$	IF NOT($condition_{11}$)
OR $condition_{21}$	AND NOT($condition_{21}$)
OR $condition_{31}$	AND NOT($condition_{31}$)
THEN positive class	THEN negative class
ELSE negative class	ELSE positive class

Figure 2.3: From DNF to CNF by applying De Morgan's laws.

- **Nonlinear metrics.** They are defined as a nonlinear combination of elementary metrics. The most popular ones are *precision*, *exclusiveness*, *information content*, *entropy*, and *FMeasure* that controls the trade-off between precision and recall.

A good evaluation measure should also care about the *complexity* of a rule, often defined in terms of the number of conditions in its antecedent (rule length), such that short rules are preferred. In practice, we aim at finding rules with a good trade-off between predictive performance and complexity.

2.3.4 Hypothesis Construction

In Section 2.2, we defined a rule as a set of Boolean conditions connected through the Boolean AND operator, also known as *conjunction*. It is clear that an hypothesis description language with a single rule is not expressive enough to fully characterize most of the target concepts, even the easiest ones (Fürnkranz et al., 2014, Section 8.1).

A way to increase the power of the hypothesis language is to allow *disjunctions*. In other words, we can describe a target concept with a disjunction of conjunction also called disjunctive normal form (DNF). A DNF is equivalent to a *rule set* where rules are connected through the Boolean OR operator.

In a binary classification problem, a DNF characterizes the target positive concept. A sample that is not covered by any rule is automatically classified as negative. It means that we can negate the DNF to obtain an equivalent Boolean representation for the negative class. By applying the De Morgan's laws (look example in Figure 2.3), the DNF becomes a conjuctive normal form (CNF) where each term and Boolean operator is negated. Some algorithms like De Raedt (1992) can learn both DNF and CNF theories.

As an alternative to rule sets, it is also possible to learn rule lists (or decision lists), where consecutive rules are connected with an IF-ELSEIF-...-ELSEIF-ELSE structure. Unlike rule

Algorithm 3 LearnRuleSet $(\mathcal{D}_+, \mathcal{D}_-, \mathcal{B})$

$\mathcal{R} = \emptyset$
$\mathcal{D}_+^{cur}, \mathcal{D}_-^{cur} = \mathcal{D}_+, \mathcal{D}_-$
while $\mathcal{D}_+^{cur} \neq \emptyset$ **do**
  $r = \text{LearnBestRule}(\mathcal{D}_+^{cur}, \mathcal{D}_-^{cur}, \mathcal{B})$
  if TheoryStopCondition $(\mathcal{R}, \mathcal{D}_+, \mathcal{D}_-)$ **then**
    break
  end
  $\mathcal{R} = \mathcal{R} \cup r$
  $\mathcal{D}_+^{cur}, \mathcal{D}_-^{cur} = \text{UpdateData}(\mathcal{R}, \mathcal{D}_+^{cur}, \mathcal{D}_-^{cur})$
end
$\mathcal{R} = \text{PostProcessRules}(\mathcal{R}, \mathcal{D}_+, \mathcal{D}_-)$
return $\mathcal{R}$

sets, where each rule is applied independently from the others, in a decision list every rule depends on the previous ones. For this reason, decision lists are less interpretable than rule sets (Lakkaraju et al., 2016). In this work, we will focus on rule sets.

The separate-and-conquer algorithm. Looking at a DNF as a set of independent rules not only increases the interpretability of the hypothesis language, but it also simplifies the learning process by transforming rule learning in the task of learning multiple rules (conjunctions) instead of a unique more complex rule. The simplest and most popular algorithm to learn a set of DNF rules is described in Algorithm 3: it is known as *separate-and-conquer* or *covering algorithm*.

While there are positive samples to be covered, LearnRuleSet runs the LearnBestRule procedure, whose implementation is widely discussed in Section 2.3.2, to learn the best rule given the current set of positive and negative samples $\mathcal{D}_+, \mathcal{D}_-$. The new rule is included in the rule set and the set of positive and negative samples is updated by the UpdateData procedure. UpdateData can update either $\mathcal{D}_+$ or $\mathcal{D}_-$, or both, by removing the positive and negative records covered by r, or by assigning a different weight to the covered samples such that covered samples will have a smaller impact on the subsequent rules (W. Cohen and Singer, 1999; Gamberger and Lavrac, 2000; Weiss and Indurkhya, 2000). The algorithm can be eventually interrupted by a stop criterion, mainly to avoid overfitting. In many practical implementations, the final rule set is post-processed by applying some filtering and/or rule refinement before being returned.

Learning directly a rule set. Instead of learning multiple rules and combining them into a single DNF expression, it is also possible to learn directly a rule set.

If we extend the search space to the whole set of possible rule sets, we can in principle

design a general framework to discover DNF formulas. Unfortunately, such space is much bigger than the rule space and additional constrains on both rule and rule set size must be fixed to make the rule set learning problem tractable (Smyth and Goodman, 1992; Rijnbeek and Kors, 2010; Su et al., 2016; Dash et al., 2018). Alternatively, other methods (Rückert and Kramer, 2003; Rückert and Raedt, 2008) resort to stochastic search to deal with the computational complexity of the search problem.

Another family of approaches comes from the *instance learning* world: all the samples belonging to the target class can be considered as single rules combined into an initial rule set that is consecutively generalized. Examples of these methods are NEAR (Salzberg, 1991), RISE (Domingos, 1996), BRACID (Napierala, 2012).

The easiest approach consists in mapping a decision tree to a rule set. Indeed, every path from the root to the leaves of the tree can be seen as a conjunction and converted to a rule. Depending on the depth of the tree, a simple conversion would probably lead to complex and unreadable rules. That is why specific algorithms (Quinlan, 1987a; Chiang et al., 2001) have been designed to simplify the extracted rules by applying a post-processing step: the resulting rule set is not only simpler but also more accurate then the original tree.

From local patterns to global rule sets. The antecedent of a rule is, by definition, a Boolean expression that captures recurring patterns in the data and contains a portion of the knowledge encoded within samples. We can use such local patterns as aggregate Boolean features and use them to learn a global model. This is what LEGO approaches (Knobbe et al., 2008; Bringmann et al., 2009; Duivesteijn et al., 2012) do: as the name suggests, (a subset of) the patterns are put together to build more complex concepts. In *associative classification* methods (Liu et al., 1998, 2000; Bayardo, 1997; Jovanoski and Lavrac, 2001; Li et al., 2001; Yin and Han, 2003), local patterns are discovered via standard data mining techniques like association rule discovery, that are usually combined according to some heuristics. In recent years, there has been a new explosion of LEGO approaches, with a more involved rule selection phase, often defining an integer optimization problem, eventually relaxed, to learn the final global model (Chang et al., 2012; Wang et al., 2015; Wang and Rudin, 2015; Lakkaraju et al., 2016; Wang et al., 2017; Yang et al., 2017; Chen and Rudin, 2018; Angelino et al., 2018).

2.3.5 Overfitting and Pruning

All machine learning methods without exceptions face overfitting issues, approximating the training set too closely and loosing their ability to generalize to unseen data. This happens for several reasons: the chosen model might be too complex for the problem at hand, the training set too small, not representative enough of the true data distribution,

Algorithm 4 PostPruning $(\mathcal{D}, split\_threshold)$

$\mathcal{D}', \mathcal{D}'' = \text{SplitData}(\mathcal{D}, split\_threshold)$
$\mathcal{R} = \text{LearnTheory}(\mathcal{D}')$
$\mathcal{R} = \text{PruneTheory}(\mathcal{D}'')$
return $\mathcal{R}$

or we might simply have a noisy dataset (Hawkins, 2004).

Rule learning commonly solve overfitting by applying *pre-pruning* and *post-pruning* techniques: *pre-pruning* decides when a rule should not be specialized anymore during the rule-induction process, *post-pruning* simplifies the final set of rules at the end of the rule-induction process in order to increase their generalization power. Nobody prevents us from using both techniques.

Pre-pruning. Pre-pruning is implemented via filtering/stopping criteria. We can distinguish between *rule stopping criteria* and *theory stopping criteria*.

- *Rule stopping criteria* interrupt the specialization of a given rule (or remove the rule) when some conditions are satisfied. They are implemented within the RuleStopCondition method in Algorithm 2. Such conditions are often based on one or more coverage/precision constraints: we might, for example, decide to remove a rule that does not cover enough target samples, a rule whose accuracy is below a given threshold (Pompe et al., 1993), or with too many conditions.
- *Theory stopping criteria* interrupt the rule learning process without covering all possible target samples, preventing the discovery of a complete theory when some conditions are satisfied. They are implemented within the TheoryStopCondition method in Algorithm 3. In FOIL (Quinlan and Cameron-Jones, 1993), for example, a theory is interrupted when the best rule is below a certain threshold.

Post-pruning. Post-pruning techniques consider overfitting as an independent problem. They are widely used in decision tree algorithms (Breiman et al., 1984; Niblett and Bratko, 1987; Quinlan, 1987b), but they are also popular in rule learning (Michalski et al., 1986; Brunk and Pazzani, 1991; Bergadano, 1992). Most of the proposals in the literature can be seen as a generalization of Algorithm 4 (Pagallo and Haussler, 1990), where rules are first learned for a given subset of data and then generalized/filtered on a separate subset of data. Generalizing an existing theory is not easy: it involves *pruning operators* that remove conditions from one or more rules to increase the performance (according to a selected measure) of the whole rule set. This process is more computationally expensive than any pre-pruning technique, but it also ensures better performance. Depending on

the implementation of PruneTheory, we might have *reduced error pruning* (Brunk and Pazzani, 1991), *reduced error regrowth* (Cohen, 1993), and so on.

Other techniques to fight overfitting. *Incremental reduced error pruning* (Fürnkranz and Widmer, 1994) integrates pre and post-pruning into a unique strategy that is both effective and efficient, compensating the disadvantages of the individual techniques. Every time a new rule is learned, it is immediately pruned. This can be seen as post-pruning from the rule perspective, and pre-pruning at the theory level.

Alternatively to pruning, rule set can be post-processed by relearning individual rules (Cohen, 1995) or selecting a subset of the rules after having ranked them. In ensemble methods, it might also be useful to re-weight rules.

2.4 Discussion

In the previous sections we briefly summarized the main characteristics of rule-based methods, identifying several building blocks that, applied one after the other, produce a theory. Such theory consists of a set (or list) of rules that are used to classify new samples; theories with a limited number of short rules are considered to be interpretable and reflect the internal "reasoning" of the model in a compact way.

Unfortunately, not all rule-based methods are interpretable. Some approaches might be more interpretable than others but, in general, interpretability does not come for free: it has to be imposed during and/or after the rule learning process. Moreover, there are other factors to consider when learning a theory. As already noted in Section 2.3.4, rule set theories are known as *disjunctions of conjunctions* (Hauser et al., 2010; Wang et al., 2015), where each disjunct represents a rule. We can distinguish between *small* and *large disjuncts* depending on the number of correctly classified training samples. In particular, small disjuncts are problematic since they have a much higher error rate than the other disjuncts, with a direct impact on the accuracy of the learned theory (Weiss and Hirsh, 2000). It has also been observed that *rare cases* (samples that occur relatively infrequently in the training data) and *imbalanced data* (where the classes are not represented equally) tend to cause small disjuncts (Weiss, 1995; Jo and Japkowicz, 2004). When this happens, *data fragmentation* and *class overlaps*, that are quite common in most of the datasets, become more problematic, especially when they involve minority classes and rare samples (Napierala, 2012).

We will now proceed with a critical review of the field, focusing on the topics highlighted above, identifying open problems and interesting research directions.

2.4.1 A Critical Review of Rule Learning Methods

In rule learning, interpretability has not always been at the core of the learning process.

Data mining solutions. Many seminal rule learning methods come from the data mining community: CBA and its extensions (Liu et al., 1998, 2000), APRIORI-C (Jovanoski and Lavrac, 2001), CPAR (Yin and Han, 2003) and CMAR (Li et al., 2001), for example, use mining to identify class association rules and then choose a subset of them according to a ranking to implement the classifier. In practice, these methods suffer from a huge number of rules, which negatively impacts interpretability.

Top-down vs. bottom-up learners. Another family of approaches includes methods like AQ (Michalski et al., 1986), PRISM (Cendrowska, 1987), CN2 (Clark and Niblett, 1989), FOIL (Quinlan and Cameron-Jones, 1993), and RIPPER (Cohen, 1995), whereby *top-down* learners build rules by greedily adding the condition that best explains the remaining data, according to some criteria. *Top-down* learners are well suited for noisy data and are known to find general rules (Fürnkranz et al., 2014, Section 2.5.2). They work well for the so called *large disjuncts*, but have difficulties to identify *small-disjuncts* and rare examples (Holte et al., 1989), which are quite common in imbalanced settings. In contrast, *bottom-up* learners like GOLEM (Muggleton and Feng, 1990) and MODLEM (Grzymala-busse and Stefanowski, 2001), start directly from very specific rules (the examples themselves) and generalize them until a given criteria is met. Such methods are susceptible to noise, and tend to induce a very high number of specific rules, but are better suited for cases where only few examples characterize the target class (Fürnkranz et al., 2014, Section 2.5.2).

Hybrid approaches. Hybrid approaches such as NEAR (Salzberg, 1991), RISE (Domingos, 1996), and BRACID (Napierala, 2012) combine rule learning and instance-based learning in a hybrid classification strategy: input samples are used to learn a set of rules, but every training sample is also seen as a *maximally-specific* rule that covers by definition only that sample. Thus, they achieve better generalization, also in imbalanced settings, but still generate many rules, penalizing interpretability.

Alternative approaches. Issues caused by small disjuncts and rare examples have been tackled acting at data-level – mainly doing data augmentation – and algorithmic-level, relying on more appropriate rule evaluation measures to reduce the maximum-generality bias, modifying the classification (Grzymala-Busse et al., 2000) or selection (Nguyen and Ho, 2005) strategy to give more strength to minority rules, preferring rule ensembles with special focus on *minority classes* (classes with few samples) (Blaszczynski et al., 2010). Unfortunately, none of the previous methods can be considered interpretable.

Modern solutions. Recent work focuses on marrying competitive predictive accuracy with *high interpretability*. As noted in the last paragraph of Section 2.3.4, a popular approach is to use the output of an association rule discovery algorithm, like FPGROWTH (Borgelt, 2005) or APRIORI (Agrawal and Srikant, 1994), and combine the discovered patterns in a *small* and *compact* set (or list) of rules with high predictive performance (LEGO approach). The combination process can be formalized either as an integer optimization problem or solved heuristically, explicitly encoding interpretability needs in the optimization function. Such approaches have been successfully applied to rule lists (Klivans and Servedio, 2006; Chang et al., 2012; Wang and Rudin, 2015; Yang et al., 2017; Chen and Rudin, 2018; Angelino et al., 2018) and rule sets (Wang et al., 2015; Lakkaraju et al., 2016; Wang et al., 2017). Alternatively, rules can be directly learned from the data through an integer optimization framework (Hauser et al., 2010; Malioutov and Varshney, 2013; Goh and Rudin, 2014; Su et al., 2016; Dash et al., 2018).

2.4.2 Challenges and Opportunities

Despite the high volume of work on rule learning, many historical problems are still challenging to solve and new ones have arisen due to interpretability constraints.

In particular, both heuristic and integer-optimization based LEGO approaches underestimate the complexity and importance of finding good candidate rules (or patterns), and become expensive when the input dimensionality increases, unless some constraints are imposed on the size and support of the rules. Although such constraints favour interpretability, they have a negative impact on the predictive performance of the model. Additionally, these methods do not explicitly consider class imbalance issues: i) they take the pattern discovery process for granted and have no guarantees that the discovered patterns will be useful to generate rules that characterize the minority classes. Our novel solution to these historical problems will be proposed in the next chapter.

Other issues emerge when rule-learning models process streams of data that change with time. In this case, rules are continuously learned, removed, adapted according to several criteria as done in STAGGER (Schlimmer and Granger, 1986), FLORA (Widmer and Kubat, 1996) and more modern systems such as RUDOLF (Milo et al., 2018) and GOLDRUSH (Jarovsky et al., 2018). Discussing in detail about *incremental rule learning* is out of the scope of this work, but we think it might become a trending topic in the next few years.

CHAPTER

THREE

LEARNING INTERPRETABLE BOOLEAN RULE ENSEMBLES

For a long time, it has been thought that rule learning, and in general interpretable models are less accurate than the so called black-box models: "interpretability comes at the cost of predictive performance" is a recurrent sentence in many publications. While this is actually true for some fields like computer vision, where black-box models (especially neural networks) attain overwhelming performance, even superior to humans, the difference in performance might be negligible, if not absent, for many other data types, like structured data. However, rule learners have many critical issues too, ranging from the NP-hard computational complexity to the ability of managing minority target classes. We present LIBRE, a novel ensemble method to learn an interpretable rule-based classifier that overcomes the aforementioned limitations. LIBRE efficiently strikes the right balance between prediction accuracy, which is competitive with black-box methods, and interpretability, which is often superior to alternative methods from the literature.

3.1 Overview

We can refine the rule learning concept defined in the previous chapter by putting interpretability as an explicit constraint. According to this view, the goal of rule learning is to learn a set of rules from the training set that i) effectively predicts a given target, ii) generalizes to unseen data, iii) is interpretable, i.e., it consists in a small number of short rules. The first objective is particularly difficult to meet in presence of imbalanced data. In this case, most rule learners (and in general many machine learning methods) fail at characterizing the minority class, that is usually of primary interest. As anticipated in Section 2.4, imbalanced data often leads to additional issues that hinder the application of rule-based methods (Weiss, 2004): data fragmentation (especially in case of *small-*

disjuncts (Holte et al., 1989)), overlaps between imbalanced classes, and presence of rare examples. The challenge is to satisfy all these requirements at the same time.

The key idea in our work is to exploit the known advantages of bottom-up learners in imbalanced settings, and improve their generalization and noise-tolerance through an ensembling technique that does not sacrifice interpretability. As a result, we produce a rule-based method that is (i) *versatile* and *effective* in dealing with both balanced and imbalanced data, (ii) *interpretable*, as it produces small and compact rule sets, and (iii) *scalable* to big datasets.

Outline of the chapter. In Section 3.2, we present a real, industrial problem that motivated us to opt for rule-based methods first, and to design then a new proposal that could overcome the practical limitations we experienced with state-of-the-art approaches. In Section 3.3, we introduce background knowledge and definitions about Boolean functions, monotone Boolean functions and their main properties. This allows us to redefine the binary classification problem as the problem of finding a boundary within a "special" binary lattice space, a *partially ordered set*. Section 3.4 is the core of the chapter: first, we describe the algorithm that implements a bottom-up learner, inspired by Muselli and Quarati (2005), but with a dramatically lower computational complexity. Such algorithm, based on monotone Boolean function synthesis, is designed to learn a small number of compact binary strings, that are naturally mapped to Boolean rules. Then, we analyze the properties of this baseline algorithm, and design a new algorithm that overcomes its limitations. We finally propose LIBRE, a novel ensemble method that, unlike other ensemble proposals in the literature (W. Cohen and Singer, 1999; Friedman and Popescu, 2008; Dembczyński et al., 2010), is interpretable. Each weak learner uses our baseline, and candidate rules are combined with a simple union, to obtain a final interpretable rule set. The idea of ensembling is crucial to improve generalization, while using *bottom-up* weak learners allows to generate meaningful rules even when the target class has few available samples. In Section 3.5, we perform an extensive experimental validation indicating that LIBRE scales to large datasets, has competitive predictive performance compared to state-of-the-art approaches (even black-box models), and produces few and simple rules, often outperforming existing interpretable models. Section 3.6 concludes the chapter with a critical analysis of LIBRE, and in general rule-base methods, and some interesting future work.

3.2 A real industrial use case

The rule-based model we are going to detail in the next sections is the result of a real need we had while facing an industrial use case. Describing this project and its requirements, highlighting the challenges we faced, is functional to our discussion.

3.2.1 Context and objectives

SAP is a software corporation that develops enterprise software to manage business operations and customer relations. The project we were involved in is placed within the broad context of data-driven predictive maintenance of IT services, and, in particular, in the context of one of the major offering of SAP services, the in-memory storage solution called SAP HANA.

General context. SAP periodically collects relevant information about the monitored SAP HANA database systems and stores them into a data lake. Collected data is heterogeneous in nature: it includes *time series information* about services and hosts, such as CPU utilization, RAM availability, disk and network utilization, database and table sizes; *tabular data* regarding configuration settings and parameters; *textual data* containing notes from the SAP personnel dedicated to predictive maintenance and optimization of services, and so on. SAP provides a diagnostic service to his SAP HANA users. Such service, called *Early-Watch Alert* (EWA), monitors the database systems, processes their data, and produces one report per system. Each report can be interpreted as the status of the system that, for the sake of simplicity, can be summarized as "anomalous" or "normal". SAP engineers analyze these reports to identify and solve eventual problems.

Objectives and requirements. EWA is a monolithic, centralized anomaly detection system, that encodes years of SAP knowledge about database services maintenance into a huge piece of code, written in ABAP, a SAP proprietary language. Modifications to EWA, mostly related to the creation and or updated of checks to detect new anomalies, imply a manual intervention on the existing code. Clearly, the more changes you do, the higher is the risk of generating bugs. Moreover, EWA is unique for every customer and system. This is a consequence of the limitations above: in practice, it is not feasible to write customer-specific code snippets, as it would not scale to the thousands of SAP customers.

SAP aims at substituting EWA, with a machine learning-based EWA, trained to learn the precious knowledge encoded into the monitoring data and reports. Machine learning systems have the advantage of automatically learning from data; it also becomes possible to have separate models for separate customers. Furthermore, machine learning models can be designed to adapt to data changes, reducing the effort of manual intervention in the system (although some human operations are still required).

Additionally, SAP requires for the machine learning model to be understandable by the personnel. Indeed, EWA is an important component of the maintenance life cycle: it should detect the presence of anomalies, but also help the personnel to figure out why anomalies have been detected, simplifying the search for a solution.

3.2.2 Proposed solution

This SAP project entails evident challenges. The first issue is related to input data: reporting the details is out of the scope of this thesis, although we made a great effort to structure the huge amount of data into a format that could be used by different machine learning models. Preliminary data analysis highlighted the second, expected problem: anomalies represented only a small percentage of data. Last, but not least, interpretability was an explicit and indispensable requirement.

After several meetings with the SAP personnel, rule-based models emerged as a possible practical solution: rules are intuitive to understand and can be seen as a preliminary explanation of the reasons behind anomalies. Unfortunately, the results of several rule-based and transparent state-of-the-art models were unsatisfactory, both in terms of interpretability and predictive performance (especially compared to black-box models). Further analysis highlighted criticalities already known in the literature: top-down methods were usually able to learn general and compact rules, but faced issues to cover minority subgroups of anomalies; bottom-up methods were having the opposite problem. We report experimental results, later in the chapter, in Section 3.5.

Having found no satisfactory answers in the existing literature led us to design a new method that could provide high predictive performance, be interpretable and applicable to strongly imbalanced settings.

3.3 Preliminaries

Our methodology targets binary classification, although it can be extended to multi-class settings. For the sake of building interpretable models, we focus on Boolean functions for the mapping between inputs and labels, which are amenable to a simple interpretation.

Boolean functions can be used as a model for binary classifiers $f(\mathbf{x}) = y$, where $\mathbf{x} \in \{0,1\}^d$, $y \in \{0,1\}$. The function f induces a separation of $\{0,1\}^d$ in two subsets $\mathcal{N}$ and $\mathcal{P}$, where $\mathcal{N} = \{\mathbf{x} \in \{0,1\}^d : f(\mathbf{x}) = 0\}$ and $\mathcal{P} = \{\mathbf{x} \in \{0,1\}^d : f(\mathbf{x}) = 1\}$. We call such subsets positive and negative subsets, respectively. Clearly, $\mathcal{N} \cup \mathcal{P} = \{0,1\}^d$ corresponds to the full truth table of the classification problem. We restrict the input space $\{0,1\}^d$ to be a *partially ordered set* (*poset*): a *Boolean lattice* on which we impose a partial ordering relation, as defined next.

Definition 3.3.1. Let $\bigwedge$, $\bigvee$, $\neg$ be the AND, OR, and NOT logic operators respectively. A *Boolean lattice* is a 5 tuple $(\{0,1\}^d, \bigwedge, \bigvee, 0, 1)$. The lack of the $\neg$ operator implies that a lattice is **not** a Boolean algebra. Let $\leq$ be *a partial order relation* such that $\mathbf{x} \leq \mathbf{x}' \iff \mathbf{x} \bigvee \mathbf{x}' = \mathbf{x}'$. Then, $(\{0,1\}^d, \leq)$ is a poset, a set on which a partial order relation has been imposed.

The theory of Boolean algebra ensures that the class $\mathcal{B}_d$ of Boolean functions $f : \{0,1\}^d \to \{0,1\}$ can be realized in terms of $\bigwedge$, $\bigvee$, and $\neg$. Nevertheless, if $\{0,1\}^d$ is a Boolean lattice, $\neg$ is not allowed and only a subset $\mathcal{M}_d$ of $\mathcal{B}_d$ can be realized. The class $\mathcal{M}_d$ coincides with the collection of monotone Boolean functions. The lack of the $\neg$ operator may limit the family of functions we can reconstruct. However, by applying a suitable transformation of the input space, we can enforce the monotonicity constraint (Muselli, 2005). Then, it is possible to find a function $\tilde{f} \in \mathcal{M}_d$ that approximates $f \in \mathcal{B}_d$ arbitrarily well.

Definition 3.3.2. Let $(\mathcal{X}, \leq)$ and $(\mathcal{Y}, \leq)$ be two posets. Then, $f : \mathcal{X} \to \mathcal{Y}$ is called *monotone* if $\mathbf{x} \leq \mathbf{x}'$ implies $f(\mathbf{x}) \leq f(\mathbf{x}')$.

Definition 3.3.3. Given $\mathbf{x} \in \{0,1\}^d$, let $\mathcal{I}_m$ be the set of the first m positive integers $\{1, \ldots, m\}$. $\mathcal{T}(\mathbf{x}) = \{i \in \mathcal{I}_m : \mathbf{x}(i) = 1\}$. The inverse of $\mathcal{T}$ is denoted as $t(\mathcal{T}(\mathbf{x}, m)) = \mathbf{x}$.

Theorem 1. (Wegener, 1987) Let $\tilde{f} \in \mathcal{M}_d$ be a monotone Boolean function. Then, it exists a partially ordered set $\mathcal{A}$ such that $\tilde{f}$ can be written as: $\tilde{f}(\mathbf{x}) = \bigvee_{\mathbf{a} \in \mathcal{A}} \bigwedge_{j \in \mathcal{T}(\mathbf{a})} \mathbf{x}(j)$.

The monotone Boolean function $\tilde{f}$ is specified in *disjunctive normal form* (DNF), and is univocally determined by the set $\mathcal{A}$ and its elements. Thus, given $\mathcal{N}$ and $\mathcal{P}$, learning $\tilde{f}$ amounts to finding a particular set of lattice elements $\mathcal{A}$ defining the **boundary** separating positive from negative samples. More formally:

Definition 3.3.4. Given $\mathbf{a} \in \{0,1\}^d = \mathcal{P} \cup \mathcal{N}$, if $\mathbf{a} \leq \mathbf{x}$ for some $\mathbf{x} \in \mathcal{P}$, and $\nexists \mathbf{y} \in \mathcal{N} : \mathbf{a} \leq \mathbf{y}$, and $\exists \mathbf{y} \in \mathcal{N} : \mathbf{b} \leq \mathbf{y}$, $\forall \mathbf{b} < \mathbf{a}$, then $\mathbf{a}$ is a *boundary point* for $(\mathcal{P}, \mathcal{N})$. The set $\mathcal{A}$ of boundary points defines the *separation boundary*. If $\mathbf{a}' \not\leq \mathbf{a}''$ and $\mathbf{a}'' \not\leq \mathbf{a}'$, $\forall \mathbf{a}', \mathbf{a}'' \in \mathcal{A}, \mathbf{a}' \neq \mathbf{a}''$, then the separation boundary is *irredundant*.

In other words, a boundary point is a lattice element that is smaller than or equal to at least one positive element in $\mathcal{P}$, but larger than all negative elements $\mathcal{N}$. In practical applications, however, we usually have access to a subset of the whole space, a training set of positive and negative samples, $\mathcal{D}_+ \subseteq \mathcal{P}$ and $\mathcal{D}_- \subseteq \mathcal{N}$ respectively. The goal of the algorithms we present next is to approximate the boundary $\mathcal{A}$, given $\mathcal{D}_+$ and $\mathcal{D}_-$. We show that boundary points, and binary samples in general, naturally translate into classification rules. Indeed, let $\mathcal{R}$ be the set of rules corresponding to the discovered boundary. $\mathcal{R}(\cdot)$ represents a binary classifier: $\mathcal{R}(\mathbf{x}) = \{1$ if $\exists r \in \mathcal{R} : r(\mathbf{x}) = 1$; 0 otherwise$\}$. Then, $\mathbf{x}$ is classified as positive if there is at least one rule in $\mathcal{R}$ that is true for it.

3.4 Boolean Rule Sets

We presented a theoretical framework that casts binary classification as the problem of finding the boundary points for $\mathcal{D}_+ \subseteq \mathcal{P}$ and $\mathcal{D}_- \subseteq \mathcal{N}$. Next, we use such framework to design our interpretable classifier.

In Section 3.4.1, we first clarify some required assumptions on the input data and how to satisfy them. In Section 3.4.2, we describe a base, bottom-up method – which will be used as a weak learner later – that illustrates how to move inside the Boolean lattice to find boundary points. However, the base method does not scale to large datasets, and tends to overfit. Thus, in Section 3.4.3, we present LIBRE, an ensemble classifier that overcomes such limitations by running on randomly selected subset of features. LIBRE is interpretable as it combines the output of an ensemble of weak learners with a simple union operation. Finally, in Section 3.4.4, we illustrate a procedure to select a subset of the generated points – the ones with the best predictive performance – and reduce the complexity of the boundary.

3.4.1 Assumptions on the Input Data

We assume that the input dataset is a poset and that the function we want to reconstruct is monotone. This is ensured by applying inverse-one-hot-encoding on discretized features, and concatenating the resulting binary features, as done in Muselli (2006). Given $z \in \mathcal{I}_m = \{1, ..., m\}$, **inverse-on-hot encoding** produces a binary string $\mathbf{b}$ of length m, where $b(i) = 1$ for $i \neq z$, $b(i) = 0$ for $i = z$. More details can be found in Appendix A.1.

Example 3.4.1. Consider a dataset with two continuous features, f_1 and f_2, both taking values in the domain $[0, 100]$. Suppose that, a discretization algorithm outputs the following discretization ranges for the two features: $[[0, 40), [40, 100]]$ and $[[0, 30), [30, 60), [60, 100]]$ respectively. Once all records are discretized, we apply inverse one-hot encoding, as previously defined. For example, $f_1 = 33.1, f_2 = 44.7$ is first discretized as $f_1' = 1, f_2' = 2$, and then binarized as 01 101. In other words, each feature of a record is encoded with a number of bits equal to its discretized domain, and can have only one bit set to zero.

3.4.2 The Base, Bottom-up Method

We develop an approximate algorithm that learns the set $\mathcal{A}$ for $(\mathcal{D}_+, \mathcal{D}_-)$. The algorithm strives to find lattice elements such that both $|\mathcal{A}|$ and $|\mathcal{T}(\mathbf{a})|$, $\forall \mathbf{a} \in \mathcal{A}$ are small, translating in a small number of sparse boundary points (short rules).

Algorithm Design. In order to proceed with the presentation of our algorithm, we need the following definitions:

Definition 3.4.1. Given two lattice elements $\mathbf{x}, \mathbf{x}' \in \{0, 1\}^d$, we say that $\mathbf{x}'$ *covers* $\mathbf{x}$, if and only if $\mathbf{x}' \leq \mathbf{x}$, according to the standard ordering in the Boolean lattice.

Definition 3.4.2. Given a lattice element $\mathbf{x} \in \{0, 1\}^d$, *flipping off* the k-th element of $\mathbf{x}$ produces an element $\mathbf{z}$ such that $\mathbf{z}(i) = \mathbf{x}(i)$ for $i \neq k$ and $\mathbf{z}(i) = 0$ for $i = k$.

Algorithm 5 FindBoundary $(\mathcal{D}_+, \mathcal{D}_-)$

Set $\mathcal{A} = \emptyset$ and $\mathcal{S} = \mathcal{D}_+$
while $\mathcal{S} \neq \emptyset$ **do**
  Choose $\mathbf{x} \in \mathcal{S}$
  Set $\mathcal{I} = \mathcal{P}(\mathbf{x})$, $\mathcal{J} = \emptyset$
  FindBoundaryPoint$(\mathcal{A}, \mathcal{D}_+, \mathcal{D}_-, \mathcal{S}, \mathcal{I}, \mathcal{J})$
  Remove from $\mathcal{S}$ the elements covered by $\mathbf{a}$, $\forall \mathbf{a} \in \mathcal{A}$
end

Definition 3.4.3. Given a positive binary sample $\mathbf{x} \in \mathcal{D}_+$, we say that a flip-off operation produces a *conflict* if the lattice element $\mathbf{z}$ resulting from the flip-off is such that $\exists \mathbf{x}' \in \mathcal{D}_- : \mathbf{z} \leq \mathbf{x}'$.

Then, a boundary point is a lattice element that covers at least one positive sample, and for which a flip-off operation would produce a conflict, as defined above.

Algorithm 5 presents the main steps of our algorithm, where $\mathcal{A}$ is the boundary set, enlarged progressively, and $\mathcal{S} = \{\mathbf{s} \in \mathcal{D}_+ : \nexists \mathbf{a} \in \mathcal{A}, \mathbf{a} \leq \mathbf{s}\}$ is the set of elements in $\mathcal{D}_+$ that are not covered by a boundary point in $\mathcal{A}$. $\mathcal{I}$ is the set of indexes of the components of the current positive sample $\mathbf{x}$ that can be flipped-off, and $\mathcal{J}$ is the set of indexes that generate a conflict with $\mathcal{D}_-$ if they are flipped-off. The role of $\mathcal{I}$ and $\mathcal{J}$ will be clear later in Algorithm 6. Until $\mathcal{S}$ is not empty, an element $\mathbf{x}$ is picked from $\mathcal{S}$. Then, the procedure FindBoundaryPoint generates one or more boundary points by flipping-off the candidate bits of $\mathbf{x}$. According to Definition 3.4.2, a boundary point is generated when an additional flip-off would lead to a conflict, given Definition 3.4.3. When the FindBoundaryPoint procedure completes its operation, both $\mathcal{A}$ and $\mathcal{S}$ are updated.

Example 3.4.2. Let $\mathcal{D}_+ = \{11001\}$ and $\mathcal{D}_- = \{01101, 01101\}$. Take the positive sample 11001, for which $\mathcal{I} = \{1, 2, 5\}$ and $\mathcal{J} = \emptyset$ at the beginning. Suppose that FindBoundaryPoint flips-off the bits in $\mathcal{I}$ from left to right. Flipping-off the first bit generates $01001 \leq 01101 \in \mathcal{D}_-$. The first bit is moved to $\mathcal{J}$ and kept to 1. Flipping-off the second bit generates $10001 \leq 10101 \in \mathcal{D}_-$. Also the second bit is moved to $\mathcal{J}$. We finally flip-off the last bit and obtain 11000 that is not in conflict with any element in $\mathcal{D}_-$. 11000 is therefore a boundary point for $(\mathcal{D}_+, \mathcal{D}_-)$.

If we think about binary samples in terms of rules – the process to translate binary samples into rules is explained at the end of this section – a positive sample can be seen as a *maximally-specific rule* that covers by definition only that particular sample. Flipping-off bits is nothing more than generalizing that rule. Our goal is to do as many flip-off operations as possible before running into a conflict.

Retrieving the complete set of boundary points requires an exhaustive search, which is

expensive, restricting its application to small, low-dimensional datasets. It is easy to show that the computational complexity of the exhaustive approach is $O(n^2 2^d)$, where n is the number of *distinct* training samples, and d is the dimension of the Boolean lattice. Efficient procedures to synthetize monotone boolean functions exist in interactive learning environments (Kovalerchuk et al., 1996; Torvik and Triantaphyllou, 2002), where we are allowed to access an oracle to which we can present specific samples and get the correct label. Unfortunately, this is not the case. In this work, we propose an *approximate heuristic* for the FindBoundaryPoint procedure.

Finding Boundary Points. The key idea is to find a subset of all possible boundary points, steering their selection through a measure of their quality. A boundary point is considered to be "good" if it contributes to decreasing the complexity of the resulting boundary set, which is measured in terms of its cardinality $|\mathcal{A}|$ and the total number of positive bits $\sum_{\mathbf{a}\in\mathcal{A}} |\mathcal{T}(\mathbf{a})|$. In practice, $|\mathcal{A}|$ can be decreased by choosing boundary points that cover the largest number of elements in $\mathcal{S}$. To do this, we iteratively select the best candidate index $i \in \mathcal{I}$ according to a measure of potential coverage. Decreasing $\sum_{\mathbf{a}\in\mathcal{A}} |\mathcal{T}(\mathbf{a})|$ implies finding boundary points with low number of 1s.

Before proceeding, we define a notion of distance between lattice elements:

Definition 3.4.4. Given $\mathbf{x}, \mathbf{x}' \in \{0,1\}^d$, the *distance* $d_l(\mathbf{x}, \mathbf{x}')$ between $\mathbf{x}$ and $\mathbf{x}'$ is defined as: $d_l(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{d} |\mathbf{x}(i) - \mathbf{x}'(i)|_+$, where $|\cdot|_+$ is equal to 1 if $(\cdot) \geq 0$, 0 otherwise.

Definition 3.4.5. In the same way, we can define the distance between a lattice element $\mathbf{x}$ and a set $\mathcal{V}$ as: $d_l(\mathbf{x}, \mathcal{V}) = \min_{\mathbf{x}'\in\mathcal{V}} d_l(\mathbf{x}, \mathbf{x}')$.

Every boundary point $\mathbf{a}$ for $(\mathcal{D}_+, \mathcal{D}_-)$ has distance $d_l(\mathbf{a}, \mathcal{D}_-) = 1$; in fact, boundary points are all lattice elements for which a flip-off would generate a conflict. In the iterative selection process of the best index $i \in \mathcal{I}$ to be flipped-off, indexes having high $d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_-{}_i^0)$ are preferred, where $\mathcal{D}_-{}_i^0 = \{\mathbf{x} \in \mathcal{D}_- : \mathbf{x}(i) = 0\}$, since they are the ones that contribute most to reduce the number of 1s of a potential boundary point.

Algorithm 6 illustrates our approximate procedure, where $\mathcal{S}_i^0 = \{\mathbf{s} \in \mathcal{S} : \mathbf{s}(i) = 0\}$ and $\mathcal{D}_+{}_i^0 = \{\mathbf{p} \in \mathcal{D}_+ : \mathbf{p}(i) = 0\}$ are proxies for the potential coverage of flipping-off a given bit i. The first step of the algorithm computes, for each index $i \in \mathcal{I}$, the terms $|\mathcal{S}_i^0|$ and $|\mathcal{D}_+{}_i^0|$ indicating its potential coverage, and $d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_-{}_i^0)$. Until the set $\mathcal{I}$ is not empty, indexes inducing a unit distance to $\mathcal{D}_-$ are moved to $\mathcal{J}$. Then, we choose the best index i_{best} among the remaining indices in $\mathcal{I}$, using our **greedy heuristics**: we can chose to optimize either for the tuple $H_1 = (|\mathcal{S}_i^0|, |\mathcal{D}_+{}_i^0|, d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_-{}_i^0))$ or for the tuple $H_2 = (d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_-{}_i^0), |\mathcal{S}_i^0|, |\mathcal{D}_+{}_i^0|)$ in lexicographic order. H_1 prioritizes a lower number of boundary points, while H_2 tends to generate boundary points with fewer 1s.

Algorithm 6 FindBoundaryPoint $(\mathcal{A}, \mathcal{D}_+, \mathcal{D}_-, \mathcal{S}, \mathcal{I}, \mathcal{J})$

For each $i \in \mathcal{I}$ compute $|\mathcal{S}_i^0|, |\mathcal{D}_{+i}^0|, d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_{-i}^0)$
while $\mathcal{I} \neq \emptyset$ **do**
  Move from $\mathcal{I}$ to $\mathcal{J}$ all i with $d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_{-i}^0) = 1$
  if $\mathcal{I} = \emptyset$ **then**
    break
  end
  Choose the best index $i \in \mathcal{I}$
  Remove i from $\mathcal{I}$
  foreach $i \in \mathcal{I}$ **do**
    update $d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_{-i}^0)$
  end
end
if *there is no* $\mathbf{a} \in \mathcal{A} : t(\mathcal{J}) \geq \mathbf{a}$ **then**
  Set $\mathcal{A} = \mathcal{A} \cup t(\mathcal{J})$
end

When $\mathcal{I}$ is empty, $t(\mathcal{J})$ is added to the boundary set $\mathcal{A}$ if it does not contain already an element covering $t(\mathcal{J})$. Note that, in Algorithm 6, the distance is computed only once, and updated at each iteration. This is because only one bit is selected and removed from $\mathcal{I}$; then, $t(\mathcal{I} \cup \mathcal{J})_{new} = t((\mathcal{I} \cup \mathcal{J})_{old} \setminus \{i\})$. Formally, we apply Definition 3.4.4 exclusively for $i = i_{best}$.

Example 3.4.3. Let $\mathcal{D}_+ = \{10101, 01101, 01110\}$ and $\mathcal{D}_- = \{10110, 11010\}$. We describe the procedure for few steps and only for the first positive sample 10101. Suppose to optimize the tuple $(|\mathcal{S}_i^0|, |\mathcal{D}_{+i}^0|, d_l(t(\mathcal{I} \cup \mathcal{J})))$. For 10101 we have $\mathcal{I} = \{1, 3, 5\}$ and $\mathcal{J} = \emptyset$. At the beginning $\mathcal{S} = \mathcal{D}_+$. $|\mathcal{D}_{+1}^0| = 2, |\mathcal{D}_{+3}^0| = 0, |\mathcal{D}_{+5}^0| = 1$. $\mathcal{D}_{-1}^0 = \emptyset, \mathcal{D}_{-3}^0 = \{11010\}, \mathcal{D}_{-5}^0 = \{10110, 11010\}$. Consequently: $d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_{-1}^0) = undefined$, $d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_{-3}^0) = 2$, $d_l(t(\mathcal{I} \cup \mathcal{J}), \mathcal{D}_{-5}^0) = 1$. Bit 5 is moved to $\mathcal{J}$. Bit 1 has the higher value of $|\mathcal{D}_{+i}^0|$ and is selected as best candidate to be flipped-off. The distance is recalculated and the procedure continues until the set of candidate bits $\mathcal{I}$ is empty.

The time complexity of Algorithm 5, when it runs Algorithm 6, is $O(n^2 d^2)$. This is faster than the exhaustive algorithm, and better than the $O(n^2 d^3)$ complexity of Muselli and Quarati (2005) (the difference between our proposal and Muselli and Quarati (2005) is detailed in Appendix A.1). We also point out that most sequential-covering algorithms repeatedly remove the samples covered by the new rules, forcing the induction phase to work in a more partitioned space with less data, especially affecting minority rules, which already rely on few samples. The problem is mitigated in our solution: despite $\mathcal{S}$ cannot avoid this behavior, our heuristics keep a global and constant view of both $\mathcal{D}_-$, in the conflict detection, and $\mathcal{D}_+$, in the discrimination of the best bits to flip.

From boundary set to rules. Each element $\mathbf{a}$ of the boundary set $\mathcal{A}$ can be practically seen as the antecedent of an **if-then** rule having as target the positive class. When a binary sample $\mathbf{x}$ is presented to $\mathbf{a}$, the rule outputs 1 only if $\mathbf{x}$ has a 1 in all positions where $\mathbf{a}$ has value 1, that is if $\mathbf{a} \leq \mathbf{x}$. Then, the antecedent of the rule is expressed as a function of the input features in the original domain.

Example 3.4.4. Consider a dataset with two continuous features, f_1 and f_2, discretized as follows: $[[0, 40), [40, 100]]$ and $[[0, 30), [30, 60), [60, 100]]$ respectively. We also apply inverse-one-hot encoding as explained in Section 3.4.1 and obtain a dataset of 6-dimensional binary strings. Assuming that our algorithm outputs a boundary set $\mathcal{A} = \{01\ 100\}$, we obtain, from the boundary point, a rule as follows: the first two bits referring to feature f_1 – 01 – are mapped to "**if** $f_1 \in [0, 40)$", while the bits referring to f_2 – 100 – are mapped to "**if** $f_2 \in [30, 100]$", where the two consecutive intervals have been combined. The zeros determine the ranges in the if conditions. The final rule is therefore "**if** $f_1 \in [0, 40)$ and $f_2 \in [30, 100]$ **then** label = 1".

3.4.3 The LIBRE Method

The base approach generates boundary points by generalizing input samples, i.e., by flipping-off positive bits if no conflict with negative samples is encountered. The hypothesis underlying this procedure is that when no conflicts are found, a boundary point induces a valid rule. However, such rule might be violated when used with unseen data. Stopping the flipping-off procedure as soon as a single conflict is found has two main effects: i) we obtain very specific rules, that might be simplified if the approach could tolerate a limited number of conflicts; ii) the rules cover no negative samples in the training set and tend to overfit.

To address these issues, a simple method would be to introduce a measure for the number of conflicts and use it as an additional heuristic in the learning process. However, this would dramatically increase the complexity of the algorithm.

A more natural way to overcome such challenges is to make the algorithm directly work on (random) subsets of features; in this way, the learning process produces more general rules by construction. Randomization is a well-known technique to implement ensemble methods that provide superior classification accuracy, as demonstrated, for example, in random forests (Ho, 1998; Breiman, 2001). By using randomization, we can directly use the methodology described in the previous sections, without modifying the search procedure. The new approach – LIBRE – is an interpretable ensemble of rules that operates on a randomized subset of features.

Formally, let E be the number of classifiers in the ensemble. For each classifier $j \in \{1, \ldots, E\}$, we randomly sample k_j features of the original space and run Algorithm 5 to

produce a boundary set $\mathcal{A}_j$ for the reduced input space. $\mathcal{A}_j$ can be generated in parallel, since weak learners are independent from each other. At this point, to make the ensemble interpretable, we crucially do not apply a voting (or aggregation) mechanism to produce the final class prediction, we do a simple union instead, such that $\mathcal{A} = \bigcup_{j=1}^{E} \mathcal{A}_j$.

We note that LIBRE addresses the problems outlined above, as we show experimentally. By training an ensemble of *weak learners* that operate on a small subset of features, we artificially inflate the probability of finding negative examples. Each weak learner is constrained to run on less features not only reducing the impact of d on the execution time, but also having an immediate effect on the interpretability of the model that is forced to generate simpler rules, exactly because it operates on fewer input features.

Note that there are no guarantees that elements of $\mathcal{A}_j$ will actually be boundary points in the full feature space: weak learners have only a partial view of the full input space and might generate rules that are not globally true. Thus, it is crucial to filter out the points that are clearly far from the boundary by using the selection procedure described in the next section.

3.4.4 Producing the Final Boundary

The model learned by our greedy heuristic materializes as a set $\mathcal{A}$, which may contain a large number of elements and, in case of LIBRE, it might also contain elements that cover many negative samples. In this section, we explain how to produce a boundary set $\mathcal{A}^*$ with a good trade-off between complexity and predictive performance. This can be cast as a *weighted set cover* problem. Since exploring all possible subsets of elements in $\mathcal{A}$ can be computationally demanding, we use a standard greedy weighted set cover algorithm.

Each element $\mathbf{a} \in \mathcal{A}$ is initially assigned a weight $w(\mathbf{a}) = \alpha|\hat{\mathcal{D}}_+(\mathbf{a})| - (1-\alpha)|\hat{\mathcal{D}}_-(\mathbf{a})|$ that is proportional to the number of positive and negative covered samples, $|\hat{\mathcal{D}}_+(\mathbf{a})|$ and $|\hat{\mathcal{D}}_-(\mathbf{a})|$ respectively. The importance of the two contributions is governed by a parameter α. At each iteration, the element $\mathbf{a}$ with the highest weight is selected; if there is more than one, the element with the highest number of zeros is preferred. All samples that are covered by the selected element are removed, and the weights are recalculated. The process continues until either all samples are covered or a stopping condition is met. In imbalanced settings, α will be close to 1 to increase the strength of minority rules.

Before running the selection procedure, with the aim of speeding up execution times, we eventually apply a filtering procedure to reduce the size of the initial set to a small number of good candidates: as proposed by Gu et al. (2003), we select the top K rules according to *exclusiveness* and *local support* (defined in Table 2.1), that are more sensible than confidence and support for imbalanced settings. Then, the final boundary set is naturally mapped to a set of rules, as explained in the previous section. Figure 3.1 shows

IF mean_corpuscular_volume $\in [90, 96)$
OR gamma_glutamyl_transpeptidase $\in [20, max]$
THEN liver_disorder = True
ELSE liver_disorder = False

Figure 3.1: Example of rule set learned by LIBRE for LIVER.

an example of compact rule set learned by LIBRE for the UCI dataset LIVER. Other rule sets can be found in Appendix A.5.

3.5 Experiments

We evaluate LIBRE in terms of predictive performance, interpretability, and scalability, and compare it with other rule-based and black-box methods.

3.5.1 Experimental Settings

Datasets. We report the results for thirteen publicly available datasets from the UCI repository (Dua and Graff, 2017) and two real industrial IT datasets – proprietary of SAP. These datasets cover several domains, have different imbalance ratios, number of records and features, as summarized in Table 3.1. Some of these datasets have been used to evaluate methods for class imbalance (Van Hulse et al., 2007) and present characteristics that make them difficult to learn: overlapping classes, noisy and rare examples. All datasets have, or were transformed to have, a binary class. The SAP datasets consist of monitoring data collected across SAP HANA database systems. They have 45 features, hand-crafted by domain experts based on low-level system metrics. SAP runs a predictive maintenance system on this data and notifies customers who confirm or discard the warnings: we use these as binary labels. SAP-C is the clean version of SAP-F, where we removed records with one missing value or more.

Comparison with other methods. We compare LIBRE with two recent works: Scalable Bayesian Rule Lists (S-BRL) (Yang et al., 2017) and Bayesian Rule Sets (BRS) (Wang et al., 2017). We also report the results for a WEKA (Hall et al., 2009) implementation of RIPPER (Cohen, 1995) and MODLEM (Grzymala-busse and Stefanowski, 2001) – as representative of top-down and bottom-up approaches – and SCIKIT-LEARN (Pedregosa et al., 2011) implementations of Decision Tree (DT) (Breiman et al., 1984), Support Vector Machine with RBF kernel (RBF-SVM) (Cortes and Vapnik, 1995)), and random

Dataset	#records	#features	imbalance_ratio	target_class
ADULT	48'842	14	.23	>50k
AUSTRALIAN	690	14	.44	2
BALANCE	625	4	.08	B
BANK	45'211	17	.12	yes
HABERMAN	306	3	.26	died
HEART	270	13	.51	presence
ILPD	583	10	.28	liver patient
LIVER	345	5	.51	drinks>2
PIMA	768	8	.35	1
SONAR	208	60	.53	R
TICTACTOE	958	9	.65	positive
TRANSFUSION	748	5	.24	yes
WISCONSIN	699	9	.34	malignant
SAP-C	287'031	45	.01	crash
SAP-F	1'554'227	45	.01	crash

Table 3.1: Characteristics of evaluated datasets.

forests (RF) (Breiman, 2001). RBF-SVM and RF are selected as popular black-box models; RF is also a representative ensemble method. Other relevant methods are not publicly available (CG (Dash et al., 2018)), are only partially implemented (BRACID (Napierala, 2012)), or the implementation provided by the authors does not work properly most probably because of some programming errors (IDS (Lakkaraju et al., 2016)).

Data preprocessing. Before running RBF-SVM, we apply the SCIKIT-LEARN standard scaler to the input data to get better results. The remaining methods have no benefits from standardization in our experiments. For S-BRL and LIBRE, we apply *ChiMerge* discretization algorithm Kerber (1992) with three discretization threshold: 4, 4.6, 6; in BRS, discretization is instead controlled by an internal parameter. In both cases, discretization is optimized during training. The remaining algorithms have no explicit need for discretization. For the methods requiring binarization, we apply *one-hot encoding*, except for LIBRE that uses *inverse one-hot encoding*.

Parameter tuning. The initial set of candidate rules for S-BRL and BRS is generated by running FPGROWTH with a minimum support of 1 and a maximum mining length of 5. We also optimize BRS and S-BRL's prior hyper-parameters by cross validation. For BRS, we

run 2 chains of 500 iterations. For RIPPER, we change the number of optimization steps between 1 and 5, and activate pruning. For MODLEM, we try all available classification strategies and condition measures. For RBF-SVM, we optimize C and γ. For DT and RF, we optimize the maximum depth in $\{5, 10, 20, None\}$, we try all possible options for max_features and use a number of trees in $\{20, 50, 100\}$ for RF. For LIBRE, we vary the number of weak learners in $E \in \{5, 20, 50\}$. Each weak learner uses up to 5 features. Additionally, we try the two heuristics H_1 and H_2, as defined in Section 3.4.2, to generate rules and vary α in $\{.5, .7, .9\}$ for weighted set cover. Parameters not reported above are all fixed to recommended or default values.

Evaluation metrics. All results refer to nested 5-fold cross validation, where the same splits are used for all methods. We use F1-score to compare the predictive performance of the classifiers, as it is well-suited to evaluate the capability to characterize the target class both in balanced and imbalanced settings. For rule-based methods, we use standard metrics from the literature to evaluate the interpretability of the rule sets, namely the *number of rules* that implement a model, and the *average number of atoms per rule*. For DT, we extract the rules following the paths from root to leaves: this captures the perception of a user who looks at the tree to understand the output of the model. For S-BRL, the number of atoms in a rule is equal to the sum of the atoms in the previous rules, highlighting the fact that a user has to go through all the rules up to the one that returns the label. For all rule-based methods, we change inequalities $(<, \leq, >, \geq)$ to ranges to have a fair comparison. For example, $f_1 \geq 3$ is converted to $f_1 \in [3, max]$.

3.5.2 Experimental Results

Predictive performance evaluation. Table 3.2 shows the means and standard deviations of the F1-score for the tested algorithms (best results in bold) and the rank of their average performance. We additionally report the results for LIBRE when it is constrained to generate at most 3 rules (LIBRE 3).

If we look at the average rank, LIBRE emerges as the best method, beating both RBF-SVM and RF, demonstrating its versatility in both balanced and imbalanced settings. LIBRE 3 is better than the other rule-based competitors, despite being constrained to generate at most 3 rules. DT, MODLEM, S-BRL and RIPPER show similar performance, even though MODLEM is usually worse for balanced settings. BRS is the worst method.

If we focus more on the single datasets, we notice that, except for AUSTRALIAN, HEART, and SONAR, LIBRE obtains consistently the highest F1-score. In BANK, ILPD, and TRANSFUSION the gap between LIBRE and the closest competitor is significant; the gap is even larger in comparison to alternative rule learners. For the remaining datasets, the differences with the competitors are less pronounced but still significant. In particular,

Dataset	RBF-SVM	RF	DT	RIPPER	MODLEM	S-BRL	BRS	LIBRE	LIBRE 3
ADULT	.62(.01)	.68(.01)	.68(.01)	.59(.02)	.66(.01)	.68(.01)	.61(.01)	**.70(.01)**	.62(.01)
AUSTRALIAN	.83(.02)	**.86(.02)**	.84(.02)	.85(.02)	.68(.28)	.82(.03)	.83(.03)	.84(.03)	.84(.03)
BALANCE	.03(.07)	.00(.00)	.01(.03)	.00(.00)	**.16(.04)**	.00(.00)	.00(.00)	**.16(.08)**	.14(.06)
BANK	.46(.01)	.50(.01)	.50(.01)	.44(.04)	.50(.03)	.50(.02)	.32(.05)	**.55(.01)**	.44(.01)
HABERMAN	.24(.10)	.26(.07)	.36(.08)	.38(.07)	.40(.07)	.17(.21)	.07(.06)	**.41(.04)**	**.41(.04)**
HEART	.78(.06)	**.79(.07)**	.71(.01)	.73(.09)	.39(.31)	.74(.05)	.70(.09)	.77(.06)	.75(.02)
ILPD	.47(.02)	.44(.08)	.42(.10)	.20(.11)	.48(.08)	.14(.13)	.09(.08)	**.54(.06)**	.52(.04)
LIVER	.58(.08)	.58(.07)	.56(.10)	.59(.04)	.58(.07)	.54(.03)	.61(.05)	.60(.07)	**.63(.06)**
PIMA	.61(.04)	.63(.04)	.60(.01)	.60(.03)	.38(.18)	.61(.07)	.03(.03)	**.64(.05)**	**..64(.05)**
SONAR	.81(.04)	**.83(.05)**	.75(.05)	.77(.08)	.70(.06)	.76(.05)	.69(.06)	.79(.03)	.76(.04)
TICTACTOE	.99(.01)	.99(.01)	.97(.01)	.98(.01)	.55(.10)	.99(.01)	.99(.01)	**1.0(.00)**	.68(.04)
TRANSFUSION	.41(.07)	.35(.06)	.35(.05)	.42(.10)	.42(.08)	.05(.10)	.04(.05)	**.49(.12)**	**.49(.12)**
WISCONSIN	**.95(.02)**	**.95(.01)**	.91(.04)	.94(.02)	**.95(.01)**	.94(.02)	.88(.03)	**.95(.01)**	.93(.02)
SAP-C	.93(.02)	.93(.01)	.85(.03)	.86(.02)	.88(.01)	.90(.01)	.68(.03)	**.95(.02)**	.72(.03)
SAP-F	-	-	-	-	-	.81(.02)	-	**.89(.03)**	.68(.04)
Avg Rank	4.0(1.8)	3.1(1.9)	5.5(1.9)	5.3(1.7)	5.0(2.8)	5.3(2.3)	7.3(2.5)	**1.5(0.9)**	4.0(2.6)

Table 3.2: F1-score (st. dev. in parenthesis).

BALANCE seems to be problematic for most of the tested methods: only the bottom-up learners MODLEM and LIBRE discover useful patterns. Additionally, RIPPER, BRS and S-BRL do not learn anything helpful to predict the positive class in ILPD; MODLEM performs considerably better. From a deeper analysis, it emerges that ILPD is an imbalanced dataset with overlapping classes: rules learned by LIBRE have an error rate close to 50% on the training set, which is a consequence of the class imbalance. RIPPER does not learn these rules, whereas BRS and S-BRL does not include such rules in the final set even when they are in the set of candidate mined rules. With SAP-C, LIBRE 3 performs better than BRS but limiting the number of rules to 3 causes a significant drop in F1-score w.r.t. LIBRE. The situation is different for SAP-F, the version of the dataset with missing values. From Table 3.1, SAP-F is more than five times bigger than SAP-C, indicating that missing values are not a negligible problem in real scenarios. A method that runs without additional pre-processing is thus truly desirable. Only LIBRE and S-BRL fit this requirement automatically treating missing values as special values, while RIPPER, BRS, and MODLEM natively manage missing values for categorical features only, but require an additional pre-processing for continuous features. Despite the huge number of missing values, results for LIBRE are comparable to other rule learners when executed on SAP-C.

Dataset	DT	RIPPER	MODLEM	S-BRL	BRS	LIBRE	LIBRE 3
ADULT	287.8(6.5)	21.4(5.2)	4957.8(36.3)	71.4(2.1)	10.0(3.3)	14.0(2.1)	**3.0(0.0)**
AUSTRALIAN	4.0(0.0)	3.8(1.2)	86.6(3.2)	5.8(0.7)	**1.8(0.4)**	2.4(1.4)	2.2(0.7)
BALANCE	48.0(12.5)	0.0(0.0)	76.5(4.6)	0.0(0.0)	**1.0(0.0)**	9.0(3.0)	**1.0(0.0)**
BANK	545.4(18.3)	9.0(1.8)	3722.6(25.5)	61.2(5.5)	4.8(1.2)	15.0(1.1)	**2.0(0.6)**
HABERMAN	37.4(4.13)	**1.0(0.0)**	73.6(2.9)	5.4(1.9)	**1.0(0.0)**	1.6(0.8)	1.6(0.8)
HEART	45.6(9.1)	2.8(0.7)	50.6(2.9)	5.8(0.7)	**2.4(0.5)**	10.6(3.0)	2.8(0.4)
ILPD	80.6(30.2)	**1.0(0.6)**	128.2(7.8)	4.8(0.7)	**1.0(0.0)**	4.4(2.3)	2.2(0.4)
LIVER	84.4(15.2)	1.4(0.8)	98.4(1.6)	4.0(0.6)	**1.0(0.0)**	3.4(1.9)	2.8(0.4)
PIMA	84.8(43.1)	2.4(2.4)	151.8(7.6)	8.4(0.5)	**1.0(0.0)**	1.6(1.0)	1.6(1.0)
SONAR	15.0(9.3)	3.6(1.4)	48.8(1.6)	3.2(0.7)	**1.0(0.0)**	6.6(1.2)	1.1(0.2)
TICTACTOE	60.4(5.2)	10.6(1.6)	25.8(1.6)	12.2(1.2)	9.0(1.1)	9.0(1.1)	**3.0(0.0)**
TRANSFUSION	100.2(48.4)	1.8(0.4)	125.8(6.1)	4.4(0.8)	**1.0(0.0)**	1.2(0.4)	1.2(0.4)
WISCONSIN	31.4(5.5)	5.0(0.6)	29.2(1.9)	7.0(1.1)	5.0(0.6)	4.2(0.7)	**3.0(0.0)**
SAP-C	622.4(51.9)	19.3(3.6)	3944.5(18.8)	47.7(4.4)	20.2(3.5)	13.0(2.4)	**3.0(0.0)**
SAP-F	-	-	-	56.4(4.6)	-	17.5(5.2)	**3.0(0.0)**
Avg Rank	5.9(0.9)	3.3(1.3)	6.5(0.9)	4.8(0.8)	**1.7(1.1)**	3.1(1.1)	**1.7(0.8)**

Table 3.3: #rules (st. dev. in parenthesis).

Dataset	DT	RIPPER	MODLEM	S-BRL	BRS	LIBRE	LIBRE 3
ADULT	9.3(0.0)	4.4(0.3)	4.3(0.1)	87.0(3.2)	**3.3(0.1)**	7.8(1.0)	6.5(0.7)
AUSTRALIAN	**2.0(0.0)**	2.4(0.3)	2.3(0.1)	7.1(1.0)	3.5(0.3)	4.4(1.8)	4.4(1.3)
BALANCE	4.4(2.9)	0.0(0.0)	3.5(0.0)	0.0(0.0)	4.0(0.0)	**2.1(0.0)**	**2.1(0.0)**
BANK	9.5(0.0)	3.0(0.2)	3.0(0.0)	89.0(7.8)	3.2(0.4)	4.7(0.5)	**2.0(0.1)**
HABERMAN	4.6(3.3)	**1.8(0.4)**	2.2(0.1)	3.7(1.0)	3.2(0.7)	2.1(0.3)	2.1(0.3)
HEART	6.2(0.3)	**2.1(0.3)**	2.3(0.1)	8.1(1.3)	3.3(0.2)	7.7(0.7)	6.1(2.7)
ILPD	8.5(1.6)	**2.1(1.3)**	**2.1(0.0)**	4.4(0.4)	2.8(0.4)	3.3(1.5)	3.0(0.6)
LIVER	8.7(1.0)	**1.3(0.4)**	2.1(0.1)	3.0(0.3)	3.4(0.5)	2.5(1.0)	**1.3(0.4)**
PIMA	6.9(2.5)	2.4(0.5)	**2.1(0.1)**	6.3(0.8)	3.6(0.5)	2.5(0.7)	2.5(0.7)
SONAR	3.8(1.5)	2.1(0.2)	**1.4(0.0)**	8.2(2.1)	4.0(0.0)	3.7(1.0)	2.2(0.8)
TICTACTOE	6.7(0.1)	**2.1(0.2)**	3.5(0.0)	21.8(1.6)	3.5(0.1)	3.8(0.8)	3.0(0.0)
TRANSFUSION	6.9(2.5)	2.8(0.4)	**2.3(0.0)**	3.8(0.6)	3.0(0.6)	2.8(0.7)	2.8(0.7)
WISCONSIN	6.1(0.4)	**2.0(0.2)**	2.2(0.1)	5.9(1.0)	3.3(0.3)	3.2(1.0)	2.8(1.0)
SAP-C	15.2(1.0)	3.8(0.3)	3.4(0.1)	75.4(4.0)	3.9(0.4)	3.3(0.2)	**3.0(0.1)**
SAP-F	-	-	-	85.6(9.7)	-	4.7(0.3)	**4.2(0.2)**
Rank	5.8(1.6)	**2.3(1.4)**	**2.3(1.0)**	6.2(1.0)	4.2(1.3)	3.7(1.5)	2.5(1.7)

Table 3.4: #atom (st. dev. in parenthesis).

Interpretability evaluation. Next, using Tables 3.3 and 3.4, we perform a function level evaluation as defined in Section 1.3.4, and measure interpretability in terms of quantity and size of the rules: models with a low number of short rules are simple, hence more interpretable. Rules have been manually checked and their interpretability assessed by human experts only for the industrial SAP datasets. In our analysis, we also refer to Table 3.2, to measure the trade-off that exists between interpretability and predictive performance. We highlight in bold the most interpretable results.

On average, in terms of number of rules, LIBRE is better than RIPPER, a top-down learner that is known to generate very general rules. LIBRE overcomes the limitations of bottom-up learners like MODLEM, that is, by contrast, the worst method together with DT. S-BRL is competitive for small datasets, but the number of rules increases considerably for bigger datasets like ADULT, BANK, and SAP. Overall, BRS generates compact rule sets, with only one rule for half of the tested datasets. However, we should also notice that, except for LIVER, these are the same datasets that give F1-score close to zero. LIBRE 3 outperforms other methods and produces the most compact rule sets for the three larger datasets, with a small impact on predictive performance.

Table 3.4 shows that all the tested methods have a similar average number of atoms. Only S-BRL has issues when the number of rules is significant (like in ADULT, BANK, and SAP datasets): as a matter of fact, in rule lists every rule depends on the previous ones, and the number of atoms easily explodes. LIBRE 3 is the second best method; LIBRE comes immediately after with an average rule size of 3.7 atoms.

F1-score vs Interpretability. From the previous analysis, LIBRE, BRS and RIPPER emerge as the most interpretable rule-based methods. When LIBRE is constrained to output at most 3 rules, it can keep up with the competitors without a significant drop in performance in most of the tested datasets. In this section, we explicitly show that LIBRE can generate more compact rule sets than the competitors, still giving superior predictive performance. We compare the rule sets leading to the best F1-scores for RIPPER and BRS (we also report S-BRL) with a few configurations for LIBRE: as a general rule of thumb, LIBRE learns both interpretable and accurate rule sets when it is trained with as many estimators as possible, with few input features per estimator.

In Figure 3.2, we report the average number of rules as a function of the F1-score: points at the bottom-right side of each plot are preferable since they correspond to compact and high predictive rule sets. Results refer to four datasets: ILPD, LIVER, PIMA and TRANSFUSION. The plots on the remaining datasets can be found in the Appendix A.4. As we see, LIBRE can be tuned to score higher than the competitors with a complexity that is same if not lower. This is mainly done by varying the maximum allowed number of rules, but other parameters have a considerable impact on the final

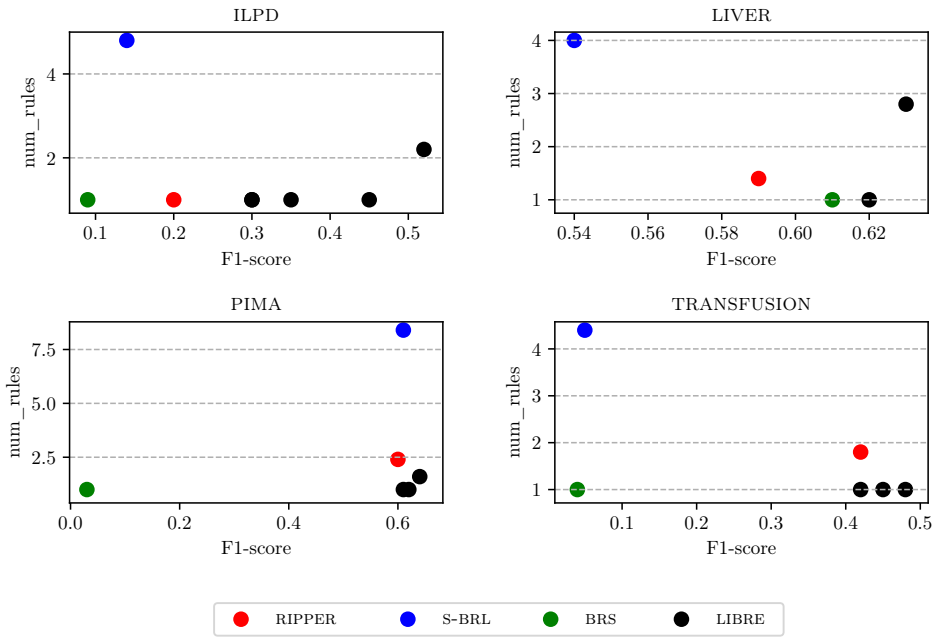


Figure 3.2: Interpretable rule sets for ILPD, LIVER, PIMA and TRANSFUSION.

performance too. In Figure 3.2, we observe that rule sets with the same number of rules have different performance, precisely because they contain different rules, learned with different parameters.

Scalability evaluation. Table 3.5 shows the training time for LIBRE and three representative rule-based competitors – RIPPER, MODLEM, BRS – on synthetic balanced datasets with 10 features and a varying number of records: from 10'000 to 1'000'000. For each configuration, we randomly generate the dataset 3 times and report the average training time and standard deviation. All methods are tested with their default parameters and run sequentially. For LIBRE, the time refers to one weak learner, which is also a good approximation for the computing time of E parallel weak learners. The symbol "-" identifies *out-of-memory* errors. In Appendix A.3, we also analyze the impact of the number of features and the class imbalance ratio on the training time.

MODLEM and BRS fail with an out-of-memory error with 500'000 and 1'000'000 records datasets. They also show much higher run times for smaller datasets w.r.t. RIPPER and LIBRE, that are instead able to complete their training in a few minutes also for the large datasets. Note that each weak learner in LIBRE works with $\mathcal{D}_+$ and $\mathcal{D}_-$ that consist of *distinct records*: although the original dataset has millions of entries, the number of binary records processed by the algorithm is much lower, especially when the

#records	RIPPER	MODLEM	BRS	LIBRE
10'000	1(0)	14(0)	144(1)	5(0)
100'000	7(3)	2457(89)	2994(304)	44(5)
500'000	39(25)	-	-	209(7)
1'000'000	101(31)	-	-	399(8)

Table 3.5: Average training time in seconds (st. dev. in parenthesis).

number of input features of each weak learner is relatively low. We also point out that, for practical applications where interpretability is needed, it is convenient to limit the number of features and train a bigger ensemble with more learners to quickly generate understandable rules.

3.6 Conclusion

Model interpretability has recently become of primary importance in many applications. In this work, we focused on the task of learning a set of rules which specify, using Boolean expressions, the classification model. We devised a practical method based on monotone Boolean function synthesis to learn rules from data. Our approach uses an ensemble of bottom-up learners that generalizes better than traditional bottom-up methods, and that works well for both balanced and imbalanced scenarios: the bottom-up weak learners learn rules that are specific enough to cover samples from the minority class, but also short and general by construction, because they operate on different small subsets of features. Discovered rules are combined with a simple union to produce an interpretable rule-set. Interpretability needs are also easily encoded in the rule generation and selection procedure that produces short and compact rule sets.

Our experiments show that LIBRE strikes the right balance between predictive performance and interpretability, often outperforming alternative approaches from the literature.

CHAPTER

FOUR

DISENTANGLED REPRESENTATION LEARNING

In the previous chapters, we focused on the interpretability of machine learning models. We implicitly assumed that input data and, more specifically, the features taken as input by the models are interpretable. When this is not true, even the simplest transparent model can be incomprehensible. In view of the considerations above, interpretability of data representations becomes central in our discussion on machine learning interpretability. In some cases, it is feasible to rely on feature engineering techniques and human experts to hand-craft understandable and useful features. In other cases, it might be more difficult, if not impossible, owing to the complexity and high-dimensionality of the data, lack of experts, resources, or any other reason. Ideally, we would like to have some tools to automatically learn features from data, to which we can assign easily interpretable meanings. This is, in short, the goal of *disentangled representation learning*. Disentangled representations are not only meaningful, but also permit to use simpler models to solve downstream tasks, for the benefit of the whole predictive chain.

4.1 Overview

Disentangled representation learning is located within the immense field of *representation learning* (also known as *feature learning*), that replaces the traditional, manual feature engineering for complex data-types, such as images and video, where it is usually hard to define specific features.

The discussion about what constitutes a good representation has lead to a series of properties, summarized in Bengio et al. (2013), on which many researchers and practitioners seem to agree. All these properties, including *interpretability*, are satisfied by

disentangled representations, that permit us to extend our work on interpretable machine learning from models to data representations. We remind the reader that transparency, as defined in Section 1.3.3, is strongly dependent on the input data.

Many approaches aimed at learning disentangled representations rely on generative models to learn a representation of the input observations that reflects the true generative, statistically independent ground-truth factors. Those factors, corresponding to semantically meaningful, hence interpretable, concepts, are assumed to generate the observations.

Recent works have made precious theoretical contributions to formally define disentangled representations, creating a connection with symmetry transformation in physics (Higgins et al., 2018a), and identifying consistency and restrictiveness as building blocks of disentanglement (Shu et al., 2020). This is essential to solve several controversial points and accelerate the progress on disentangled representation learning.

Outline of the chapter. Section 4.2 is a short introduction to representation learning. It is useful to motivate the work on disentangled representations. In Section 4.3, we present the two most complete and general theoretical framework available today to define disentangled representations, together with popular approaches and measures to evaluate the quality of learned representations in terms of disentanglement. In Section 4.4 we report many applications that benefit from disentangled representation. Section 4.5 concludes the chapter with a discussion on the current status of the research on disentangled representation learning and introduces the next chapter.

4.2 Representation Learning in a Nutshell

The performance of machine learning algorithms strongly depends on data representation. Feature engineering techniques have always been used not only as a solution to the *curse of dimensionality* problem, but also to build better data representations, by incorporating human and prior knowledge into the model learning process. However, these approaches are time consuming and expensive, as they often require the intervention of experts on the target topics. *Representation learning* is the answer to the real need to make machine learning models less dependent on the input data representation, such that they can learn useful features by themselves. More formally, representation learning aims at "learning data representations such that it is easier to extract useful information when building classifiers or other predictive tasks" (Bengio et al., 2013).

Barlow et al. (1989) advocate, probably for the first time, the utility of learning data representations consisting of statistically independent factors. They denote this concept as *factorial code discovery*. According to their intuition, representations where each

dimension is independent of any of the others would have lead to significant benefits in terms of downstream tasks. Along this line of work, we also mention the contributions on independent component analysis (ICA) (Comon, 1994; Hyvärinen and Pajunen, 1999). Deep learning approaches, in particular generative models, such as generative adversarial networks (Goodfellow et al., 2014) and auto-encoders (Kingma and Welling, 2014; Rezende et al., 2014), are other suitable choices to automatically learn representations from data. Despite statistical independence is recognized by many as a desirable property of good representations, there are other interesting characteristics at the core of *disentangled representations*, that will be investigated in the next sections.

4.2.1 What Makes a Representation Good

Theoretically, any representation that is useful to simplify subsequent downstream tasks can be considered a good representation. Bengio et al. (2013) identify a list of additional properties that good representations should satisfy.

- **Expressiveness.** A representation is *expressive* if, given a reasonable size, it allows to distinguish among a high number of different input configurations. For example, a one-hot encoding representation of size N encodes N configurations at most; a representation of the same size, learned by an auto-encoder, encodes many more configurations and it is therefore more expressive.
- **Abstractness.** A representation is *abstract* if it captures high-level concepts. Deep architectures and, in general, models that build features through composition of functions are known to learn more abstract concepts than flat architectures (Mahendran and Vedaldi, 2015).
- **Invariance.** Abstract representations also ensure another important property: they are invariant to local changes. For example, a representation that successfully encodes in its dimensions the identity of the corresponding observations should be invariant to rotations, or any other transformation that do not change the identity of the object.
- **Interpretability.** A representation is interpretable if we can i) easily assign semantically meaningful concepts to its dimensions, ii) understand how changes in one or more dimensions are reflected by the corresponding observation.

4.2.2 Disentangling Factors of Variation

Disentangled representations satisfy all the above properties. Intuitively, a *disentangled representation* is aligned to some independent, conceptually meaningful, factors of variations, also known as ground-truth factors, that are assumed to generate the input observations. Similarly to factorial analysis and ICA, disentangled representations consider

independence as a useful property. The ground-truth factors can be seen as high-level features, invariant to local changes and interpretable. More precisely, a disentangled representation is an optimal representation, in that it is the most compact, general, and interpretable one.

To the best of our knowledge, the notion of disentangled representations has been introduced for the first time in Bengio et al. (2013). Before that, several works, especially from the computer vision community, tried to learn specific factors of variations from data, without explicitly mentioning disentanglement. Only in the last few years, learning disentangled representations has become one of the hottest topic in machine learning.

4.3 Defining and Evaluating Disentangled Representations

Disentangled representation learning is currently in its infancy, hence continuously evolving. The first contributions to the topic were based on intuitive definitions of disentanglement, some approaches explored paths that have been later proved to be unsuccessful. Evaluating these models without necessarily having access to ground-truth labels is still an open problem. The practical utility of disentangled representation has to be fully proved, yet, although recent works show evidence of that.

In the following sections, we present one of the first attempts to formally define disentangled representations. Higgins et al. (2018a) use group theory to show that there exists a formal connection between *symmetry transformation* and disentangled representations. We also make some references to the work of Shu et al. (2020) that, few years later, propose a theoretical framework that defines disentangled representations in terms of two distinct concepts: *consistency* and *restrictiveness*. We postpone the discussion about whether and how disentangled representations can be learned to the next chapter.

4.3.1 Symmetry Transformations and Disentanglement

In physics, symmetry transformations, often described by means of group theory, are at the basis of many fundamental theories that explain the world dynamics. Intuitively, a symmetry of a physical system or object is a transformation that preserves certain properties of the object. This is also the mathematical definition of *invariance*. If we consider, for example, a general object such as a stone, we know that it maintains its physical properties even if we move it from one point to another or rotate it; therefore, we say that the stone is invariant with respect to translation and rotation. Equivalently, translation and rotation are symmetries for the stone.

The importance of symmetries. The study and research on structure-preserving transformations has lead to important findings in our history. In 1871, the chemist

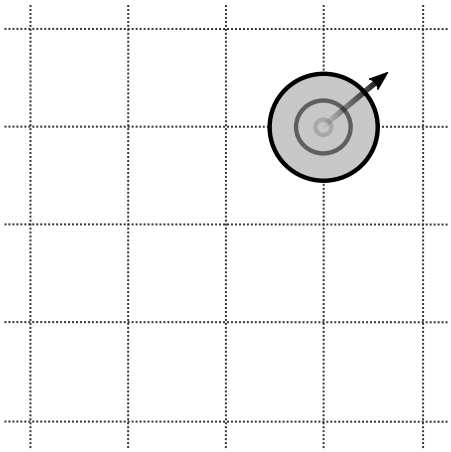


Figure 4.1: Example of a toy world, where an object moves vertically and horizontally within a grid. The object can also change its size. It can be seen as a simplified version of DSPRITES.

and inventor Dmitri Ivanovich Mendeleev, noting gaps within the periodic table of chemical elements, predicted the properties of missing elements, that he named eka-boron, eka-aluminium, eka-manganese, and eka-silicon. These properties emerged to be good predictors of the properties of scandium, gallium, technetium, and germanium, respectively. Symmetry transformations have been extended to many other domains, demonstrating their utility and generalization power.

A toy world. Higgins et al. (2018a) are the first to relate symmetry transformations with disentangled representations. In machine learning, we do not deal with physical objects but with observations. We consider images as observations, as they allow a better understanding of the covered topics. In Figure 4.1, we report an example of a toy world, a simplified version of DSPRITES (Higgins et al., 2017a), where a circular object is allowed to move vertically and horizontally within a grid, and change its size. The horizontal and vertical position, together with the size, constitute a *world state*. Every change in at least one of the three dimensions affects the world state. Clearly, when the size of the object increases or decreases, it has no effects on the identity of the object, that remains a circle, it does not even affect its position. The same happens if the horizontal or vertical positions change. Those actions that modify a certain aspect of the world state, while keeping the other fixed are called *disentangled group actions*. Disentangled group actions can be further decomposed into *disentagled sub-groups*: in this case, horizontal position, vertical position, and size.

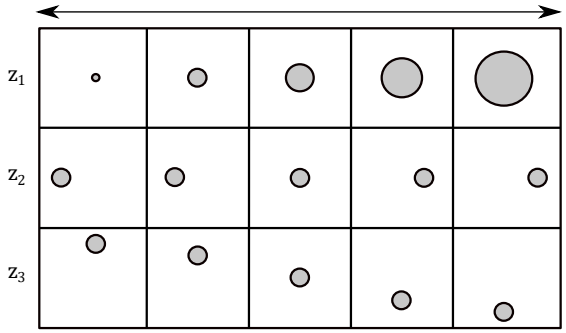


Figure 4.2: Toy world example. Latent traversal of a representation that is disentangled with respect to the following sub-group action decomposition: size (z_1), horizontal position (z_2), vertical position (z_3).

From world states to observations. Let assume that the toy world described above is the result of a generative process that transforms world states into observations. At this point, we can try to learn a representation of the world observations by training a model, for example an auto-encoder. In certain circumstances, we might be able to find a mapping from the observations to the learned representation, and vice-versa, that preserves the original disentangled group actions. If the mapping exists, it gives a disentangled representation.

Definition 4.3.1. (Higgins et al., 2018a) A vector representation is called a *disentangled representation* with respect to a particular *decomposition of a symmetry group into sub-groups*, if it decomposes into independent sub-spaces, where each subspace is affected by the action of a single sub-group, and the actions of all other sub-groups leave the subspace unaffected.

Definition 4.3.1 clarifies doubts emerged from previous work, where it was not clear how many dimensions should encode a given sub-group. According to the definition above, there is no constraint on the dimensionality of each sub-group. Nevertheless, although disentanglement is linked to a specific sub-group decomposition, disentangled group actions can generally be decomposed into different disentangled sub-groups, with different degrees of utility depending on the target task. It can reasonably be assumed that there exists a "natural" decomposition we are interested in, that can be discovered through active perception. Then, we implicitly assume to search for a representation that is aligned with such a natural sub-group decomposition. Figure 4.2 shows a representation that is disentangled with respect to horizontal position, vertical position and size.

Even in the simplest setting where each dimension of the learned representation affects only a specific property of the observations, it might be complex to describe *how* it happens. We would like to increase, for example, the value of the dimension encoding

the horizontal position of our toy world state and see the observation moving from left to right. In general, we should be able to easily understand how changes in one sub-group influence the resulting observation. This is possible if we impose a linearity constraint.

Definition 4.3.2. (Higgins et al., 2018a) A vector representation is called a *linear disentangled representation* with respect to a particular *decomposition of a symmetry group into sub-groups*, if it is a disentangled representation with respect to the same group decomposition, and the actions of all the sub-groups on their corresponding sub-spaces are linear.

4.3.2 Disentanglement and Group Theory

The previous section was an informal introduction to symmetry transformation and their connection to disentangled representations. Here, we use group theory to provide a more formal definition of the concepts seen above: disentangled group and sub-group actions, disentangled and linear disentangled representations.

Disentangled group and sub-group action. Let us assume to have a group action $\cdot : \Gamma \times X \to X$, where X is the space of observations (world space) and Γ is the group that acts on X. Any action changes the world state and generates a new object in the same world space. Let us further assume that the group Γ can be decomposed into n sub-groups Γ_i: $\Gamma = \Gamma_1 \times \cdots \times \Gamma_n$. The corresponding sub-group action is denoted by $\cdot_i$. Then, the action is disentangled with respect to the sub-group decomposition of Γ, if there exists a decomposition $X = X_1 \times \cdots \times X_n$ and actions $\cdot_i : \Gamma_i \times X_i \to X_i, i \in \{1, \cdots, n\}$ such that:

$$(\gamma_1, \cdots, \gamma_n) \cdot (x_1, \cdots, x_n) = (\gamma_1 \cdot_1 x_1, \cdots, \gamma_n \cdot_n x_n). \tag{4.1}$$

In other words, every property i of the observation space is invariant to the action of $\Gamma_j, j \neq i$, and it is affected only by the corresponding sub-group action Γ_i.

Disentangled representation. Let us denote by W the set of world states. Observations O are the result of a generative process $g : W \to O$. An external agent learns a representation Z through the mapping $e : O \to Z$. Let $f = e \circ g$ represent the composition of e and g, where we first obtain observations from world states through g, and the agent learns its own representations from the observations through e. Let us also consider a group of symmetry transformations associated with W via the group action $\cdot : \Gamma \times W \to W$. The goal is to find an equivalent group action $\cdot : \Gamma \cdot Z \to Z$. This is possible if:

$$\gamma \cdot f(w) = f(\gamma \cdot w), \forall \gamma \in \Gamma, w \in W. \tag{4.2}$$

The equation above means that applying the group action γ on the representation learned by the model $f(w)$ is equivalent to applying the same group action γ to the world

state w first, and then applying the mapping f. Hence, f is an *equivariant* map. When Equation 4.2 is satisfied, given a group decomposition into n sub-groups $\Gamma = \Gamma_1 \times \cdots \times \Gamma_n$, the agent's representation is disentangled with respect to this decomposition if the action $\cdot$ on Z is disentangled according to Equation 4.1.

Linear disentangled representation. A group action that transforms the disentangled sub-space linearly considerably increases the interpretability of the learned representation and is beneficial for downstream tasks. In this case, we talk about *linear disentangled representations*. The reasoning and definitions are similar as before, with the only difference that i) both the group action and the sub-group actions in Equation 4.1 are assumed to be linear, and ii) the group action that acts on Z is linear, too.

4.3.3 Consistency, Restrictiveness, Disentanglement

An alternative theory on disentangled representation decomposes disentanglement into two distinct concepts: consistency and restrictiveness (Shu et al., 2020). People who are not familiar with group theory might find the next definitions more intuitive.

Let us refer to our toy world as in Figure 4.1 and let assume that a generative model is trained to learn a representation that reflects the original world state. If the first dimension z_1 of the learned representation is disentangled with respect to the size, a visual inspection of z_1 would show the following: i) when z_1 is fixed, the size of the generated object does not change, ii) when z_1 varies, only the size of the generated object changes. According to Shu et al. (2020), these two properties are known as *generator consistency* and *generator restrictiveness*. If this is true for the horizontal and vertical position as well, then the generative model has learned a (fully) disentangled representation. We point out that consistency and restrictiveness are aligned with the mathematical definition of invariance that motivated the theory by Higgins et al. (2018a).

More formally, let us assume that target observations are the result of a true generative model $h^* = (p^*(s), g^*, e^*)$, where $p^*(s)$ represents the distribution over the ground-truth factors s, g^* is a generator function that maps the ground-truth factor state to the observation space, e^* is the encoder function that maps from the observation to the ground-truth factor state space. The goal is to learn a model $h = (p, g, e)$, whose learned representation z disentangles the true ground-truth factor state of h^*.

Generator consistency. Let I denote a set of indices ($\setminus I$ is its complement) and p_I denote the generating process that consists in sampling $z_I \sim p(z_I)$ once, and conditionally sampling $z_{\setminus I}, z'_{\setminus I} \sim p(z_{\setminus I}|z_I)$ in a i.i.d. fashion. We say that z_I is consistent with s_I if:

$$\mathbb{E}_{p_I} d(e^*_I \circ g(z_I, z_{\setminus I}), e^*_I \circ g(z_I, z'_{\setminus I})) = 0, \tag{4.3}$$

where e^*_I is the encoder of h^* restricted to the indices I, $d(\cdot, \cdot)$ is an appropriate distance function, and $\circ$ is the function composition operator. Equation 4.3 states that, when z_I is fixed, the representation retrieved by the true encoder model e^*, restricted to indices I, is invariant to changes in $z_{\setminus I}$.

Generator restrictiveness. Let I denote a set of indices ($\setminus I$ is its complement) and $p_{\setminus I}$ denote the generating process that consists in sampling $z_{\setminus I} \sim p(z_{\setminus I})$ once, and conditionally sampling $z_I, z'_I \sim p(z_I|z_{\setminus I})$ in a i.i.d. fashion. We say that z_I is restricted to s_I if:

$$\mathbb{E}_{p_{\setminus I}} d(e^*_{\setminus I} \circ g(z_I, z_{\setminus I}), e^*_{\setminus I} \circ g(z'_I, z_{\setminus I})) = 0, \tag{4.4}$$

where $e^*_{\setminus I}$ is the encoder of h^* restricted to the indices $\setminus I$. Equation 4.4 states that, when $z_{\setminus I}$ is fixed, the representation retrieved by the true encoder model e^*, restricted to indices $\setminus I$, is invariant to changes in z_I.

Generator disentanglement. Let I denote a set of indices, $C(I)$ and $R(I)$ are Boolean functions representing consistency and restrictiveness for the set I, respectively. We say that z_I disentangles s_I if:

$$D(I) = C(I) \wedge R(I). \tag{4.5}$$

Equations 4.3 to 4.5 can equivalently be expressed from the perspective of a learned encoder. If we want z to disentangle s, Equation 4.5 has to be true for all the ground-truth factors. It should be noted that, similarly to Higgins et al. (2018a), i) there is no constraint on the number of dimensions assigned to each ground-truth factor; ii) all the definitions above implicitly define consistency, restrictiveness and disentanglement with respect to a "natural" set of ground-truth factors.

4.3.4 Evaluating Disentanglement

For many years, *visual inspection* has been the only way to evaluate disentangled representations. *Latent traversal* used to be a popular technique to visualize the effects that each portion of the learned representation has on the corresponding observation. In the simplest settings, every dimension is varied, in turn, within a range of values, keeping the others fixed. The purpose is to investigate whether the inspected dimension disentangles one or more ground-truth factors. In Figure 4.3, we report an example of latent traversal for a content-style disentanglement model (Kingma et al., 2014) that has learned a representation that separately encodes class digit and writing style from MNIST observations. When the portion of the representation related to the content is varied, the digit changes but the style remains coherent.

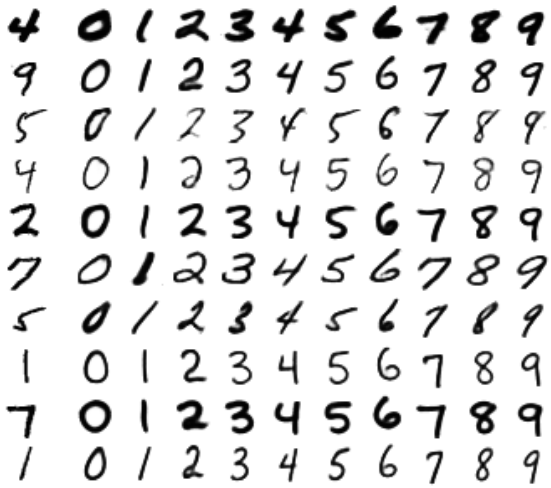


Figure 4.3: Visual Inspection of a content-style disentanglement method (Kingma et al., 2014) trained on MNIST. If we move from left to right, we observe the effects that the content portion of the learned representation has on the corresponding observation.

Even though visual inspection techniques provide a qualitative evaluation of disentanglement methods, it is evident that they require human intervention to verify disentanglement and suffer from scalability issues. These issues pushed the researchers to design practical metrics to quantitatively measure disentanglement. Despite several attempts to propose standard evaluation frameworks, there is not a globally accepted procedure yet. In this section, we report some of the most popular disentanglement metrics, with a special focus on their properties and limitations.

Beta score. The idea behind the *beta score* (Higgins et al., 2017a) is that a disentangled representation should consist of latent dimensions that are both independent and interpretable. Here, interpretability means that every latent can be easily associated to one of the true, semantically meaningful factors that generate the input observations. Assuming a full access to the generative process, the authors propose to fix a random ground-truth factor k and sample two mini batches of observations from the corresponding generative model. The encoder is then used to obtain a learned representation from the observations (with the ground-truth factor k in common). The dimension-wise absolute difference between the two representations is computed and a simple linear classifier is used to predict the fixed ground-truth factor k. This is repeated $batch\_size$ times and the accuracy of the predictor becomes the disentanglement metric score. In a perfectly disentangled representation, a fixed factor of variation corresponds to an absolute difference of zero: the linear classifier would then learn to map the index of the zero value to the index of the ground-truth factor.

Factor score. Despite being intuitive, the beta score has several weaknesses: i) it uses a linear classifier that, by definition, can learn only a linear mapping between multiple dimensions of the learned latent space and the single ground-truth factors, ii) it is susceptible to hyper-parameter tuning of the linear classifier, iii) it returns a maximum score even when only $K-1$ out of the K ground-truth factors are disentangled. For such reasons, Kim and Mnih (2018) suggest a few modifications to the beta score, ending up with a new disentanglement measure: the *factor score*. The factor score uses a majority vote classifier to predict the index of the fixed ground-truth factor based on the index with the smallest variance. A majority vote classifier does not require any parameter tuning; moreover, it assigns the lowest variance to a given factor, avoiding the failure mode of the beta score. In a perfectly disentangled representation, the fixed factor of variation corresponds to a variance of zero in the dimension encoding that factor.

SAP - Separated Attribute Predictability. With the beta score as a reference, Kumar et al. (2018) design a new disentanglement measure that does not require any additional parameter tuning for the classifier. They propose the *Separated Attribute Predictability* – SAP score – that is computed from a score matrix, where each entry is the linear regression or classification score (in case of discrete factors) of predicting a given ground-truth factor with a given dimension of the learned representation. The SAP score is the average difference of the prediction error of the two most predictive learned dimensions for each factor. For regression, they use the R^2 score obtained with fitting a line minimizing the linear regression error. A high SAP score indicates that each ground-truth factor is mostly captured by one latent dimension only.

MIG - Mutual Information Gap. The *mutual information gap* – MIG score – (Chen et al., 2018) is computed as the average, normalized difference between the highest and second highest mutual information of each ground-truth factor with the dimensions of the learned representation. Similarly to the SAP score, the MIG score needs to know the ground-truth factors, without having explicit access to the generative process. It reaches its maximum when each latent dimension is informative of a single ground-truth factor.

DCI Disentanglement. Eastwood and Williams (2018) define three properties for representations: *disentanglement*, *completeness* and *informativeness*. i) Each of the K dimensions of the learned representation is assigned a *disentanglement score* $D_i = 1 - H_K(P_{i\cdot})$, where $H_K(P_{i\cdot}) = -\sum_{k=0}^{K-1} P_{ik} \log_K P_{ik}$ denotes the entropy and $P_{ij} = R_{ij} / \sum_{k=0}^{K-1} R_{ik}$ denotes the probability that the latent dimension i is predictive about ground-truth factor j; R is an importance matrix. Then, $D_i = 1$ if dimension i is significant to predict a single ground-truth factor; $D_i = 0$ if it is equally important to predict every ground-truth factor. ii) Each ground-truth factor is assigned a *completeness*

score $C_j = 1 - H_D(P_{\cdot j})$, where $H_D(P_{\cdot j}) = -\sum_{d=0}^{D-1} P_{dj} \log_D P_{ij}$. $C_j = 1$ if ground-truth factor j is predictable via a single dimension of the learned representation; $C_j = 0$ if it is equally predictable by any dimension of the learned representation. iii) The *informativeness score* is computed for the whole representation and measures the amount of information useful to capture the ground-truth factors: it is computed as the prediction error of predicting the ground-truth factors from the learned representation.

Modularity and Explicitness. Similarly to Eastwood and Williams (2018), Ridgeway and Mozer (2018) identify three properties for representations: *modularity*, *compactness*, and *explicitness*, corresponding to the previously defined disentanglement, completness, and informativeness. However, they argue that compactness (completness) should not characterize disentangled representations. A representation is *modular* if each dimension of the learned representation depends on at most one ground-truth factor. Ridgeway and Mozer (2018) propose to measure modularity as the average normalized squared difference of the mutual information of the factor of variations with the highest and second-highest mutual information with a dimension of the learned representation. A representation is *explicit* if it is easy to predict a factor of variation. To compute explicitness, they train a one-versus-rest logistic regression classifier to predict the ground-truth factor of variation and measure its ROC-AUC.

Other measures. The work by Ridgeway and Mozer (2018) has been recently extended by Do and Tran (2020), who suggest three relevant properties for disentangled representations: *informativeness*, *separability*, and *interpretability*. A learned representation is *informative* about the corresponding observation if their mutual information is high. Two dimensions of the learned representation are *separable* if they do not share common information about the corresponding input observation. A dimension of the learned space is *interpretable* with respect to a given ground-truth factor if it only contains information about it. Zhou et al. (2020) are the first to propose an unsupervised disentanglement measure that does not require access to the ground-truth factors and exploits topological similarity of conditional sub-manifolds in the learned representation.

4.4 Practical Applications

The value of disentangled representations is not only due to their interpretability. Learning a representation that reflects the generative factors of input data offers several practical advantages: many works argue that disentangled representations simplify downstream tasks and improve training efficiency (Bengio et al., 2013; Locatello et al., 2019b, 2020b, 2019a; van Steenkiste et al., 2019; Locatello et al., 2020a); their benefits are also evident in transfer learning (Higgins et al., 2017b, 2018b; Laversanne-Finot et al., 2018; Gondal

et al., 2019; Dittadi et al., 2020), fair machine learning (Creager et al., 2019; Locatello et al., 2019a), and many other tasks. Disentangled representations have also shown superior robustness against adversarial attacks (Willetts et al., 2020).

4.4.1 Simplifying Downstream Tasks

Disentanglement has always been recognized as a desirable property in learned representations to solve downstream tasks (Bengio et al., 2013; Peters et al., 2017; Tschannen et al., 2018; Locatello et al., 2019b, 2020b, 2019a; van Steenkiste et al., 2019; Locatello et al., 2020a). Intuitively, a representation where ground-truth factors are encoded into different dimensions should make subsequent tasks, which depend on those factors, easier. Indeed, disentangled representations have been shown to be more sample-efficient (Higgins et al., 2018b), less sensitive to nuisance variables (Lopez et al., 2018), and to lead to better generalization performance (Higgins et al., 2017b; Eastwood and Williams, 2018; Steenbrugge et al., 2018).

In the easiest setting, most of the disentanglement metrics from the literature can be seen as a proxy for the performance of a simple downstream task, i.e., matching each latent dimension with the corresponding ground-truth factor. However, there is relatively few work on their applicability to more difficult, real problems. Recently, Barrett et al. (2018) have found out that standard deep neural network architectures struggle to solve complex reasoning tasks. More specifically, the authors were focusing on *abstract reasoning*, where the goal is to learn abstract relationships among multiple objects (for example images, or objects within images), and generalize such knowledge to new settings, a task that humans naturally do during their life. Motivated by those findings, Steenbrugge et al. (2018) and van Steenkiste et al. (2019) showed that models with high disentanglement scores lead to better downstream performance in less time and with less samples, compared to models with the same architecture but specifically trained to solve the abstract-reasoning tasks.

4.4.2 Transfer Learning

The goal of transfer learning is to exploit the knowledge learned while solving a given problem, to solve a different, but related one. If a model learns a disentangled representation, we expect the transfer process to be easier and more effective: intuitively, the ground-truth factors should be useful to solve many different tasks.

One of the first methods that tried to exploit disentangled representation for transfer learning is DARLA (Higgins et al., 2017b), a reinforcement learning method where the agent is able to deal with changes to the input distribution (*domain adaptation*) by learning a disentangled, task-independent representation of the world. More recent works

(Locatello et al., 2019b, 2020b) show experimental evidence that, at least for synthetic datasets, models with high disentanglement scores are better suited for transfer learning. Gondal et al. (2019); Dittadi et al. (2020) focus on more realistic settings by proposing both a synthetic and realistic dataset: those works show that model's parameters learned from a simulated dataset are effective in the real dataset, although a direct transfer of representations does not seem to work.

4.4.3 Increasing Fairness in Predictions

When machine learning is applied to solve specific tasks, sensitive protected attributes like race, gender, skin color and so on, can have undesirable impacts in many different ways. The algorithm design might cause discrimination towards protected groups, the data itself might be biased either because the collection process is biased or simply because society is unfair. The goal of fair machine learning is to learn a predictor that solves a given task without being biased by some sensitive factors. Every time a sensitive factor has a negative impact, in terms of discrimination, for the target task solved by a machine learning model, we say that such model is not fair with respect to that sensitive factor. In the literature, we have several notions of fairness: demographic parity (Calders et al., 2009), individual fairness (Dwork et al., 2012), equal opportunity (Hardt et al., 2016), and causal reasoning concepts (Kilbertus et al., 2017).

Locatello et al. (2019b) suggest a possible connection between disentangled representations and fair machine learning. Disentanglement methods learn representations with low mutual information among different dimensions encoding the ground-truth factors; this should encourage the downstream predictor to look only at the dimensions that are directly linked to the target task, discarding information coming from sensitive factors. Locatello et al. (2019a) propose a theorem asserting that even a perfect classifier might be unfair (in terms of demographic parity), when sensitive and non-sensitive factors are entangled. Then, they show empirically that general purpose representations are highly unfair; there is rather a strong correlation between disentanglement and fairness on downstream tasks for a wide range of datasets. Creager et al. (2019) propose a flexibly-fair version of VAE, where sensitive attributes are encoded into independent portions of the latent space. Again, they show that such representations lead to more fair results, in terms of demographic parity.

4.4.4 Higher Robustness against Adversarial Attacks

Deep learning models are subjected to adversarial attacks. In short, the attacker's goal is to apply the smallest amount of distortion (often not perceivable by humans) to input observations, attempting to fool the model to return a different output from what it is expected. Unfortunately, generative models are vulnerable to adversarial

attacks too (Tabacof et al., 2016; Gondim-Ribeiro et al., 2018; Kos et al., 2018): the most effective attack mode consists in distorting the input such that its latent representation is reconstructed in a different target. This attack is problematic in applications where the latent representation is used to perform one or more downstream tasks.

We expect disentangled representations to be more robust to this kind of attacks: indeed, the imperceptible distortions applied by the attacker on the input observations can be seen as small local changes that should not have a considerable impact on the latent representation, if it is disentangled. Willetts et al. (2020) is the first work who finds a strict relation between disentanglement and robustness. They first show that β-TCVAE is more robust to adversarial attacks than a standard VAE. Besides, the authors introduce a hierarchical version of β-TCVAE that further increases robustness.

4.4.5 Other Applications

A large body of work on disentanglement originates from the computer vision community, where the purpose is to disentangle content from pose in images (Tenenbaum and Freeman, 2000; Hinton et al., 2011; Zhu et al., 2014; Reed et al., 2014; Kulkarni et al., 2015; Yang et al., 2015), and pose from motion in videos (Goroshin et al., 2015; Denton and Birodkar, 2017; Villegas et al., 2017; Hsieh et al., 2018; Li and Mandt, 2018) in order to better solve specific tasks like segmentation and classification. Such kind of data allows visual inspection of the latent space and a direct understanding of the impact of each dimension of the learned representation, in the absence of appropriate disentanglement metrics. In this context, full disentanglement of all possible ground-truth factors has not been considered of primary importance.

4.5 Discussion

The concept of interpretable representations opened a whole research field, of which disentangled representations are only a part. In this chapter, we discussed the status of theoretical formalization of disentangled representations, evaluation methods and applications. We have not gone into detail about how to obtain disentangled representations, yet, but it is already clear that there is huge space for improvements and contributions.

If we focus on the problem statement, we realize that the definition itself of disentangled representations is immature: assuming that observations are the result of a transformation of independent ground-truth factors might be unrealistic. In practical scenarios, it is usually more convenient to consider a certain degree of causal relation among ground-truth factors and design new frameworks that support *causal disentanglement*. Let consider the example in Figure 4.4, where a swinging pendulum generates a shadow when hit by a light source. Clearly, the position and size of the shadow are not independent, as

Figure 4.4: A swinging pendulum that creates a shadow when hit by a light source.

they are affected by both the pendulum angle and light position. We expect the definition of disentangled representations to evolve in the next years. Causal disentanglement would be a natural extension, considering that there are already works that relax the independence assumption and try to learn causal relations among ground-truth factors (Yang et al., 2020). This would necessarily lead to alternative evaluation measures that are coherent with the new definitions.

There is also a lot to investigate about the benefits of disentanglement. Most of the experiments that justify the employment of disentangled representations to solve downstream tasks, simplify transfer learning, and so on, are carried out either for simple, synthetic datasets where the generative process is known, or realistic datasets with well-defined ground-truth factors. Results are promising, but there is room for improvements.

CHAPTER

FIVE

AN IDENTIFIABLE DOUBLE VAE FOR DISENTANGLED REPRESENTATIONS

A large part of the literature on disentangled representation learning focuses on variational autoencoders (VAE). Recent developments demonstrate that disentanglement cannot be obtained in a fully unsupervised setting without inductive biases on models and data. However, Khemakhem et al. (2020) suggest that employing a particular form of factorized prior, conditionally dependent on auxiliary variables complementing input observations, can be one such bias, resulting in an identifiable model with guarantees on disentanglement. Working along this line, we propose a novel VAE-based generative model with theoretical guarantees on identifiability. We obtain our conditional prior over the latents by learning an optimal representation, which imposes an additional strength on their regularization. We also extend our method to semi-supervised settings. Experimental results indicate superior performance with respect to state-of-the-art approaches, according to several established metrics proposed in the literature on disentanglement.

5.1 Overview

In disentangled representation learning, the main assumption is that high-dimensional observations $\mathbf{x}$ are the result of a transformation, possibly nonlinear, applied to a low dimensional latent variable of independent generative factors, called *ground-truth factors*, capturing semantically meaningful concepts. Input observations can be thought of as the result of a probabilistic generative process, where latent variables $\mathbf{z}$ are first sampled from a prior distribution $p(\mathbf{z})$, and then the observations $\mathbf{x}$ are sampled from $p(\mathbf{x}|\mathbf{z})$. The goal is to learn a representation of the data that captures the generative factors. As

defined in Section 4.3, multiple dimensions of the learned representation can theoretically be mapped to a single factor. Nevertheless, in the simplest settings each dimension of a disentangled representation refers to a single factor of variation.

We closely examine deep generative models to learn disentangled representations, and in particular those based on variational autoencoders (VAE). A well known theoretical result asserts that disentanglement is essentially impossible in a fully unsupervised setting, without inductive biases on models and data (Locatello et al., 2019b). However, inducing a disentangled structure into the latent space where $\mathbf{z}$ lies is feasible by incorporating auxiliary information $\mathbf{u}$ about the ground-truth factors in the model. The type and amount of supervision define different families of disentanglement methods, often classified as supervised, semi-supervised, and weakly-supervised. In most of these methods, the auxiliary variables $\mathbf{u}$ become an integral part of the latent space. However, recent work (Khemakhem et al., 2020) indicates that there are alternative strategies to benefit from auxiliary information, such as using it to impose a structure on the latent space. In their proposal, this is done by learning a prior distribution on the latent space, where the crucial aspect is that this is conditioned on auxiliary information $\mathbf{u}$ that is coupled with every input observations. Under mild assumptions, it is possible to show that such form of conditioning implies model identifiability, allowing one to recover the original ground-truth factors and therefore providing principled disentanglement.

In this work, we propose a novel generative model that, like Khemakhem et al. (2020), uses a conditional prior and has theoretical identifiability guarantees. We show that our method naturally imposes an optimality constraint, in information theoretic terms, on the conditional prior: this improves the regularization on the function that maps input observations to latent variables, which translates in tangible improvements of disentanglement in practice. Since assuming to have access to auxiliary variables for each input observations, both at training and testing time, is not practical in many applications, we also propose a semi-supervised variant of our method.

Outline of the chapter In Section 5.2, we give background on model identifiabiliy and disentanglement; then, in Section 5.3, we report a detailed overview of VAE-based disentanglement methods using a unified notation. Our focus is on the role of the regularization term. We introduce a distinction between *direct matching approaches*, in which ground-truth factors are directly matched to the latent space, and *indirect matching approaches*, where a prior distribution over the latents is used to structure the learned latent space. In Section 5.4, we propose a new method to learn disentangled representations – that we call *Identifiable Double* VAE (IDVAE) since its ELBO can be seen as a combination of two variational autoencoders – that is identifiabile, in theory, and that learns an optimal conditional prior, which is truly desirable in practice. We

additionally propose a semi-supervised version of IDVAE to make our method applicable also when auxiliary information is available for a subset of the input observations only. In Section 5.5, we design an experimental protocol that uses four well-known datasets, and established disentanglement metrics. Finally, by comparing our method to several several state-of-the-art competitors, we demonstrate that IDVAE achieves superior disentanglement performance across most experiments.

5.2 Preliminaries

5.2.1 Model Identifiability and Disentanglement

Let $\mathbf{x} \in \mathbb{R}^n$ be some input observations, which are the result of a transformation of independent latent ground-truth factors $\mathbf{z} \in \mathbb{R}^d$ through a function $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^n$. Then, we have that $\mathbf{x} = \mathbf{f}(\mathbf{z}) + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a Gaussian noise term: $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$, and independent of $\mathbf{z}$. Let consider the following generative model:

$$p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{z}), \tag{5.1}$$

where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ is a vector of model parameters, $p_{\boldsymbol{\theta}}(\mathbf{z}) = \prod_{i=1}^{d} p_{\boldsymbol{\theta}}(z_i)$ represents the factorized prior probability distribution over the latents and $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ is the conditional distribution to recover $\mathbf{x}$ from $\mathbf{z}$. The decoder function $\mathbf{f}(\mathbf{z})$ (plus the noise $\boldsymbol{\epsilon}$) determines the way $\mathbf{z}$ is transformed into $\mathbf{x}$ within $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$.

Assume to observe some data $\mathcal{D} = \{\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(N)}\}$ generated by $p_{\boldsymbol{\theta}^*}(\mathbf{x}, \mathbf{z}) = p_{\boldsymbol{\theta}^*}(\mathbf{x}|\mathbf{z})p_{\boldsymbol{\theta}^*}(\mathbf{z})$, where $\boldsymbol{\theta}^*$ are the true, but unknown parameters. The goal is to learn $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ such that:

$$p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = p_{\boldsymbol{\theta}^*}(\mathbf{x}, \mathbf{z}). \tag{5.2}$$

When Equation 5.2 holds, it is then possible to recover the generative ground-truth factors. Unfortunately, by observing $\mathbf{x}$ alone, we can estimate the marginal density $p_{\boldsymbol{\theta}}(\mathbf{x}) \approx p_{\boldsymbol{\theta}^*}(\mathbf{x})$, but there are no guarantees about learning the true generative model $p_{\boldsymbol{\theta}^*}(\mathbf{x}, \mathbf{z})$. This is only feasible for models satisfying the following implication:

$$\forall(\boldsymbol{\theta}, \boldsymbol{\theta}') : p_{\boldsymbol{\theta}}(\mathbf{x}) = p_{\boldsymbol{\theta}'}(\mathbf{x}) \implies \boldsymbol{\theta} = \boldsymbol{\theta}'. \tag{5.3}$$

When Equation 5.3 holds, the estimated and the true marginal distribution match, and their parameters match too. Then, the model is **identifiable** (Khemakhem et al., 2020) and, as a consequence, it allows one to recover the latent ground-truth factors and obtain a disentangled representation:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = p_{\boldsymbol{\theta}'}(\mathbf{x}) \implies p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = p_{\boldsymbol{\theta}'}(\mathbf{x}, \mathbf{z}). \tag{5.4}$$

A practical goal is to aim for model identifiability *up to trivial transformations*, such as permutation and scaling; as long as ground-truth factors can be identified, their order and scale is irrelevant.

5.2.2 Connections with Independent Component Analysis (ICA)

In the traditional formulation of both linear and nonlinear ICA, the goal is to recover the ground-truth factors (more commonly called *sources* or *components* in this context) from a set of observations. The independent components (except perhaps one) must be non-Gaussian. The function f, called mixing function, that maps sources to observations, is usually deterministic, although noisy formulations are also possible.

Linear ICA In linear ICA, the mixing function f is invertible and linear, that is each observation is a linear combination of the sources. Thus, the generative model in Equation 5.2 is an under-specified linear system. For simplicity, the number of observations and components is often assumed to be the same: f is then a square *mixing matrix* and finding f^{-1} is equivalent to find its inverse. A known result of linear ICA (Comon, 1994) is that the statistical independence of the sources is a sufficient condition for identifiability (or in alternative terminology the "sources can be separated") up to a permutation and scaling of the sources (Hyvärinen and Oja, 2000).

Nonlinear ICA When the mixing function f is nonlinear, we talk about nonlinear ICA. Although the remaining settings and final goal are the same, nonlinear ICA is much more complicated than linear ICA: the statistical independence of the sources is no longer a sufficient condition to recover them. For the first time, Hyvärinen and Pajunen (1999) demonstrate that i) when the space of mixing functions is not limited and ii) observations are i.i.d., there exists an infinity of solutions, and indeterminacies are not trivial as in the linear case. Thus, additional constraints are needed to resolve these ambiguities.

The first attempts towards identifiability used to put strong, often unrealistic, constraints on f. More successful approaches rely on the relaxation of i.i.d. hypothesis: a relevant amount of work focuses on time-series, where it is possible to exploit the temporal structure of the data (Harmeling et al., 2003; Sprekeler et al., 2014; Hyvärinen and Morioka, 2016, 2017). Practical solutions for non-temporal data are also proposed (Tan et al., 2001; Harmeling et al., 2003; Almeida, 2004) but principled and more general approaches are pretty recent (Hyvärinen et al., 2019; Khemakhem et al., 2020).

5.3 VAE-based Disentanglement Methods

A large body of work on disentangled representations is based on generative models, especially VAE-based approaches (Kingma and Welling, 2014; Rezende et al., 2014).

5.3.1 Unsupervised Disentanglement Learning

A standard VAE learns the parameters of Equation 5.1 by introducing an inference model $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ to derive an ELBO as follows:

$$\mathcal{L}_{\text{VAE}} = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})] - \text{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})), \tag{5.5}$$

where, by abuse of notation, we write $\mathbf{x}$ in place of $\mathbf{x}^{(i)}$. This avoids clutter in the presentation of VAE-based models, but, clearly, the marginal log-likelihood is composed of a sum of such ELBO terms, one for each observation $\mathbf{x}^{(i)}$ (Kingma and Welling, 2014).

The distribution $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ has the role of a decoder, whereas $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ can be seen as an encoder, and it is generally assumed to be a factorized Gaussian with a diagonal covariance matrix. Both distributions are parameterized with neural networks, with parameters $\boldsymbol{\theta}$ and variational parameters $\boldsymbol{\phi}$. The prior $p(\mathbf{z})$ is generally a factorized, isotropic unit Gaussian.

The first term of Equation 5.5 relates to the *reconstruction* of the input data using latent variables sampled from the variational approximation of the true posterior. The second one is a *regularization* term, which pushes the approximate posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ to match the prior on the latent space. Maximizing Equation 5.5 across observations implies learning the parameters such that the reconstruction performance is high, and the regularization term is small. In other words, a VAE learns a latent representation $\mathbf{z} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ that efficiently transmit information about the input observations $\mathbf{x}$, such that the $\mathbf{x}$ can be reconstructed from $\mathbf{z}$.

Since both terms that appear in the regularization of Equation 5.5 are factorized Gaussians with diagonal covariance, one way to interpret the individual components z_i of the latent space is to view them as independent white noise Gaussian channels (Burgess et al., 2017). Clearly, when the KL term is zero, the latent channels z_i have zero capacity ($\mu_i = 0, \sigma_i = 1$) because they do not transmit useful information for the reconstruction task: this happens when the approximate posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ matches exactly the prior $p_{\boldsymbol{\theta}}(\mathbf{z})$. Figure 5.1 shows that broad posterior distributions with close means (close to zero) decrease the KL diverge with the prior. Nevertheless, a sample $\tilde{x}$ from distribution $q(z_2|x_2)$ is more likely to be confused with a sample from $q(z_1|x_1)$, with a consequent increase of the cost in terms of log likelihood. In order to increase the capacity of the latent channels and improve the reconstruction quality, the approximate posterior must deviate from the isotropic unit Gaussian prior by decreasing the overlap between channels, and reducing their variances.

The above understanding of the regularization term is at the basis of many variants of the original VAE model, that strive to increase the pressure on the regularization term, or

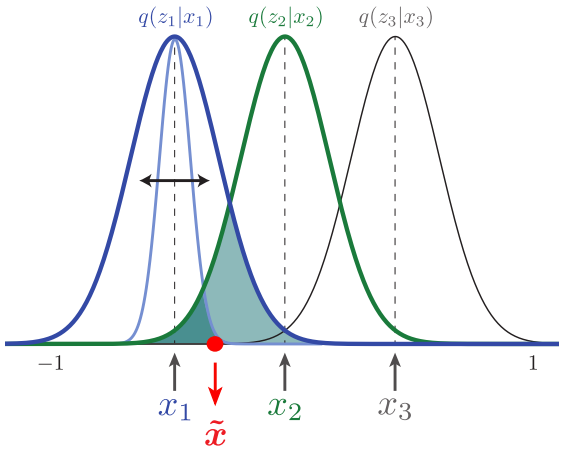


Figure 5.1: Minimizing the KL divergence increases the posterior overlaps (Burgess et al., 2017).

elements thereof, to achieve disentanglement, without sacrificing reconstruction properties too much. For example, Higgins et al. (2017a) propose β-VAE, which modifies Equation 5.5 by introducing a hyper-parameter β to gauge the pressure on the regularization term throughout the learning process:

$$\mathcal{L}_{\beta-\text{VAE}} = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})] - \beta \text{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})). \tag{5.6}$$

When $\beta > 1$, the encoder distribution $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ is pushed towards the unit Gaussian prior $p(\mathbf{z})$. In light of the discussion above, the strong penalization of the KL term in β-VAE affects the latent channel distribution, by reducing the spread of their means, and increasing their variances.

Many methods build on β-VAE (Burgess et al., 2017; Kim and Mnih, 2018; Kumar et al., 2018; Chen et al., 2018; Zhao et al., 2019; Esmaeili et al., 2020), rewriting the ELBO in slightly different ways. A generalization of the KL term decomposition proposed by Hoffman and Johnson (2016); Makhzani and Frey (2017) is the following (Chen et al., 2018):

$$\mathbb{E}_{\mathbf{x}}[\text{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))] = I(\mathbf{x};\mathbf{z}) + \text{KL}(q(\mathbf{z})||\prod_j q(z_j))) + \sum_j \text{KL}(q(z_j)||p(z_j)) \tag{5.7}$$

where $q(\mathbf{z})$ is the aggregated posterior and $I(\mathbf{x};\mathbf{z})$ is the mutual information between $\mathbf{x}$ and $\mathbf{z}$. Penalizing $I(\mathbf{x};\mathbf{z})$ can be harmful to reconstruction purposes, but enforcing a factorized aggregated posterior encourages independence across the dimensions of $\mathbf{z}$, favouring disentanglement. The dimensional independence in the latent space is encouraged by the second term, known as total correlation (TC). The third term is a

further regularization, preventing the aggregate posterior to deviate too much from the factorized prior.

Note that unsupervised VAE-based approaches approximate the data marginal distribution $p_{\boldsymbol{\theta}}(\mathbf{x})$, but there are no guarantees to recover the true joint probability distribution $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$, having access to the input observations $\mathbf{x}$ only (Khemakhem et al., 2020). Pushing the model to learn a representation with statistically independent dimensions is not sufficient to obtain full disentanglement. These considerations have been recently formalized in the *impossibility result* (Locatello et al., 2019b), but they were already known in the nonlinear ICA literature (Comon, 1994; Hyvärinen and Pajunen, 1999).

5.3.2 Auxiliary Variables and Disentanglement

In order to overcome the above limitations, a key idea is to incorporate an inductive bias in the model. The choice of the variational family and prior distribution can be one of such bias (Mathieu et al., 2019; Kumar and Poole, 2020). Alternatively, it is possible to rely on additional information about the ground-truth factors, which we indicate as $\mathbf{u} \in \mathbb{R}^m$. When auxiliary observed variables $\mathbf{u}$ are available, they can be used jointly with $\mathbf{z}$ to reconstruct the original input $\mathbf{x}$. These methods are usually classified under the semi/weakly supervised family. More specifically Shu et al. (2020) identify three commonly used forms of weak supervision: *restricted labeling* (Kingma et al., 2014; Cheung et al., 2015; Siddharth et al., 2017; Klys et al., 2018), *match/group pairing* (Bouchacourt et al., 2018; Hosoya, 2019; Locatello et al., 2020a), and *rank pairing* (Chen and Batmanghelich, 2020a,b). When all ground-truth factors are known for all the input samples, we label them as supervised disentanglement methods.

As for unsupervised counterpart, methods relying on auxiliary observed variables $\mathbf{u}$ differ in how the regularization term(s) are designed. Some approaches use a "supervised" regularization term to directly match $\mathbf{z}$ and the available ground-truth factors $\mathbf{u}$: we refer to this form of regularization as *direct matching*. An example is what here we call FULLVAE method (Locatello et al., 2020b), which optimizes the following ELBO:

$$\mathcal{L}_{\text{FULLVAE}} = \mathcal{L}_{\beta\text{-VAE}} - \gamma R_s(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}), \mathbf{u}), \tag{5.8}$$

where $R_s(\cdot)$ is a loss function between the latent and the ground-truth factors (in the original implementation it is a binary cross entropy loss). Other approaches employ a KL divergence term between the posterior and the prior over the latents: we refer to this form of regularization as *indirect matching*. In other words, direct matching methods require explicit knowledge of one or more ground-truth factors, whereas indirect matching can also use weak information about them. Shu et al. (2020) have demonstrated that indirect matching methods can enforce some properties in the latent space, leading to

what they define as *consistency* and *restrictiveness* (see Section 4.3.3). To obtain full disentanglement, a method must satisfy both properties on all the latent dimensions.

A recent work by Khemakhem et al. (2020) establishes a theoretical framework to obtain model identifiability, which is related to disentanglement. They propose a new generative model called IVAE, that learns a disentangled representation using a factorized prior from the exponential family, crucially conditioned on $\mathbf{u}$. In practical applications, the conditional prior is chosen to be a Gaussian location-scale family, where the mean and variance of each latent dimension z_i are expressed as a function of $\mathbf{u}$. Hence, it is possible to derive the following ELBO for the IVAE model:

$$\mathcal{L}_{\text{IVAE}} = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x},\mathbf{u})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})] - \beta \text{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x},\mathbf{u})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{u})). \tag{5.9}$$

In Equation 5.9, we recognize the usual structure of a reconstruction, and a regularization term. A remarkable advancement of IVAE relates to its identifiability properties: next, we present a new identifiable model to learn disentangled representations, by using an optimal factorized prior, conditionally dependent on auxiliary information. We also extend our method to deal with more realistic semi-supervised settings.

5.4 IDVAE - Identifiable Double VAE

Let $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{u} \in \mathbb{R}^m$ be two observed random variables, and $\mathbf{z} \in \mathbb{R}^d$ a low-dimensional latent variable, with $d \leq n$. Then, consider the following generative models:

$$p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}|\mathbf{u}) = p_{\mathbf{f}}(\mathbf{x}|\mathbf{z}) p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u}), \tag{5.10}$$

$$p_{\mathbf{f}}(\mathbf{x}|\mathbf{z}) = p_{\boldsymbol{\epsilon}}(\mathbf{x} - \mathbf{f}(\mathbf{z})), \tag{5.11}$$

$$p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u}) = \prod_i h_i(z_i) g_i(\mathbf{u}) \exp\left[\mathbf{T}_i(z_i)^\top \boldsymbol{\eta}_i(\mathbf{u})\right], \tag{5.12}$$

and

$$p_{\boldsymbol{\vartheta}}(\mathbf{z}, \mathbf{u}) = p_{\boldsymbol{\vartheta}}(\mathbf{u}|\mathbf{z}) p(\mathbf{z}), \tag{5.13}$$

where $\boldsymbol{\theta} = (\mathbf{f}, \mathbf{T}, \boldsymbol{\eta})$ and $\boldsymbol{\vartheta}$ are model parameters. Equation 5.10 corresponds to the process of generating $\mathbf{x}$ given the latents $\mathbf{z}$. Equation 5.11 implies that $\mathbf{x} = \mathbf{f}(\mathbf{z}) + \boldsymbol{\epsilon}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$. We approximate the injective function $\mathbf{f}$ with a neural network. Equation 5.12 is an exponential conditionally factorial distribution (Bishop, 2006), where h_i is the base measure, $g_i(\mathbf{u})$ is the normalizing constant, $\mathbf{T}_i = [T_{i,1}, \cdots, T_{i,k}]^\top$ are the sufficient statistics, and $\boldsymbol{\eta}_i(\mathbf{u}) = [\eta_{i,1}, \cdots, \eta_{i,k}]^\top$ are the corresponding parameters. The dimension of each sufficient statistic k is fixed. Equation 5.13 formalizes the additional process to obtain $\mathbf{u}$ given $\mathbf{z}$ through $p_{\boldsymbol{\vartheta}}(\mathbf{u}|\mathbf{z})$, where $p(\mathbf{z})$ is a prior over the latents, usually a factorized, isotropic unit Gaussian.

Given a dataset $\mathcal{D} = \{(\mathbf{x}^{(1)}, \mathbf{u}^{(1)}), \cdots, (\mathbf{x}^{(N)}, \mathbf{u}^{(N)})\}$ of observations generated according to Equations 5.10 to 5.13, we are interested in finding a variational bound $\mathcal{L}$ for the marginal data log-likelihood $p(\mathbf{x}, \mathbf{u})$, which we derive as follows:

$$\log p(\mathbf{x}, \mathbf{u}) = \mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}, \mathbf{u})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}, \mathbf{u})) + \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}),$$

where, by abuse of notation, we write $\mathbf{x}$ and $\mathbf{u}$ in place of $\mathbf{x}^{(i)}$ and $\mathbf{u}^{(i)}$, which we do hereafter as well.

Since the KL term is non-negative, we have the following variational lower bound: $\log p(\mathbf{x}, \mathbf{u}) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi})$. Now, we can write the ELBO, which resembles that of Equation 5.9, but includes an additional term:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x},\mathbf{u})}[\log p_{\mathbf{f}}(\mathbf{x}|\mathbf{z})] - \beta \mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}, \mathbf{u})||p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u})) + \log p(\mathbf{u}), \tag{5.14}$$

where we introduce the parameter β to gauge the pressure on the KL term. Next, focusing on the generative model in Equation 5.13, we derive the following variational lower bound for $\log p(\mathbf{u})$ in Equation 5.14, $\log p(\mathbf{u}) \geq \mathcal{L}_{\text{prior}}(\boldsymbol{\vartheta}, \boldsymbol{\psi})$:

$$\mathcal{L}_{\text{prior}}(\boldsymbol{\vartheta}, \boldsymbol{\psi}) = \mathbb{E}_{q_{\boldsymbol{\psi}}(\mathbf{z}|\mathbf{u})}[\log p_{\boldsymbol{\vartheta}}(\mathbf{u}|\mathbf{z})] - \mathrm{KL}(q_{\boldsymbol{\psi}}(\mathbf{z}|\mathbf{u})||p(\mathbf{z})), \tag{5.15}$$

By combining Equation 5.14 and Equation 5.15, we obtain:

$$\begin{aligned}\mathcal{L}_{\text{IDVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\vartheta}, \boldsymbol{\psi}) = &\underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x},\mathbf{u})}[\log p_{\mathbf{f}}(\mathbf{x}|\mathbf{z})] - \beta \mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}, \mathbf{u})||p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u}))}_{(1)} \\ &+ \underbrace{\mathbb{E}_{q_{\boldsymbol{\psi}}(\mathbf{z}|\mathbf{u})}[\log p_{\boldsymbol{\vartheta}}(\mathbf{u}|\mathbf{z})] - \mathrm{KL}(q_{\boldsymbol{\psi}}(\mathbf{z}|\mathbf{u})||p(\mathbf{z}))}_{(2)}.\end{aligned} \tag{5.16}$$

We refer the reader to Appendix B.1 for the full derivation of Equations 5.14 to 5.16. We call our method IDVAE, Identifiable Double VAE, because it can be seen as the combination of two variational autoencoders (1) and (2), with independent parameters. In principle, when we optimize the ELBO by summing across all datapoints, e.g. using a doubly stochastic approach (Titsias and Lázaro-Gredilla, 2014) and automatic differentiation, we could treat the two parts separately. However, nothing would prevent $p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u})$ and $q_{\boldsymbol{\psi}}(\mathbf{z}|\mathbf{u})$ to converge to different distributions. Thus, we further make the modeling assumption of constraining the conditional prior in (1) to be exactly the variational approximation learned in (2), which belongs to the exponential family: in our current implementation, they are both Gaussian distributions, hence we can simply do moment-matching. IDVAE is trained by alternating optimization of Equation 5.16, where we update part (2), then use the conditional prior to update the part (1), and loop.

5.4.1 Identifiability Properties

Next, we set up notations and definitions for a general theory of identifiability of generative models (Khemakhem et al., 2020), and show that IDVAE, under mild conditions, is identifiable.

Notation. Concerning the exponential conditionally factorial distribution in Equation 5.12, we denote by $\mathbf{T}(\mathbf{z})$ the vector of concatenated sufficient statistics defined as follows: $\mathbf{T}(\mathbf{z}) = [\mathbf{T}_1(z_1)^\top, \cdots, \mathbf{T}_d(z_d)^\top]^\top \in \mathbb{R}^{dk}$. We denote by $\boldsymbol{\eta}(\mathbf{u})$ the vector of its parameters defined as follows: $\boldsymbol{\eta}(\mathbf{u}) = [\boldsymbol{\eta}_1(\mathbf{u})^\top, \cdots, \boldsymbol{\eta}_d(\mathbf{u})^\top]^\top \in \mathbb{R}^{dk}$.

Definition 5.4.1. Let $\sim$ be an equivalence relation on the parameter space $\boldsymbol{\Theta}$. We say that Equation 5.1 is $\sim$-identifiable if $p_{\boldsymbol{\theta}}(\mathbf{x}) = p_{\boldsymbol{\theta}^*}(\mathbf{x}) \Longrightarrow \boldsymbol{\theta} \sim \boldsymbol{\theta}^*$.

Definition 5.4.2. Let $\sim$ be the equivalence relation on $\boldsymbol{\Theta}$ defined as follows: $(\mathbf{f}, \mathbf{T}, \boldsymbol{\eta}) \sim (\mathbf{f}', \mathbf{T}', \boldsymbol{\eta}') \Leftrightarrow \exists \mathbf{A}, \mathbf{c} : \mathbf{T}(\mathbf{f}^{-1}(\mathbf{x})) = \mathbf{A}\mathbf{T}'(\mathbf{f}'^{-1}(\mathbf{x})) + \mathbf{c}, \forall \mathbf{x} \in \mathcal{X}$, where $\mathbf{A}$ is a $dk \times dk$ matrix and $\mathbf{c}$ is a vector of dimension dk. If $\mathbf{A}$ is invertible, we denote this relation by $\sim_A$.

Definition 5.4.2 establishes a specific equivalence relation that allows to recover the sufficient statistics of our model up to a linear matrix multiplication.

Theorem 2. (Khemakhem et al., 2020) Assume we observe data sampled from $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}|\mathbf{u}) = p_{\mathbf{f}}(\mathbf{x}|\mathbf{z}) p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u})$, where $p_{\mathbf{f}}(\mathbf{x}|\mathbf{z})$ as in Equation 5.11 and $p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u})$ as in Equation 5.12, with parameters $\boldsymbol{\theta} = (\mathbf{f}, \mathbf{T}, \boldsymbol{\eta})$. Assume the following holds:

i The set $\{\mathbf{x} \in \mathcal{X} : \phi_{\boldsymbol{\epsilon}}(\mathbf{x}) = 0\}$ has measure zero, where $\phi_{\boldsymbol{\epsilon}}$ is the characteristic function of the density $p_{\boldsymbol{\epsilon}}$ defined in $p_{\mathbf{f}}(\mathbf{x}|\mathbf{z}) = p_{\boldsymbol{\epsilon}}(\mathbf{x} - \mathbf{f}(\mathbf{z}))$.

ii The function $\mathbf{f}$ is injective.

iii The sufficient statistics $T_{i,j}$ in Equation 5.12 are differentiable almost everywhere, and linearly independent on any subset of $\mathcal{X}$ of measure greater than zero.

iv Being k the dimensionality of the sufficient statistics $\mathbf{T}_i$ in Equation 5.12 and d the dimensionality of $\mathbf{z}$, there exist $dk+1$ distinct points $\mathbf{u}^0, ..., \mathbf{u}^{dk}$ such that the $dk \times dk$ matrix E defined as follows is invertible:

$$\mathbf{E} = (\boldsymbol{\eta}(\mathbf{u}^1) - \boldsymbol{\eta}(\mathbf{u}^0); \cdots; \boldsymbol{\eta}(\mathbf{u}^{dk}) - \boldsymbol{\eta}(\mathbf{u}^0)) \tag{5.17}$$

Then the parameters $\boldsymbol{\theta} = (\mathbf{f}, \mathbf{T}, \boldsymbol{\eta})$ are $\sim_A$-identifiable.

Theorem 2 (proof in Appendix B.3) guarantees a general form of identifiability for IDVAE. Under more restrictive conditions on $\mathbf{f}$ and $\mathbf{T}$, following the same reasoning of Khemakhem et al. (2020), it is also possible to reduce $\mathbf{A}$ to a permutation matrix.

Note that, in practice, all the VAE-based methods we discuss in this work are approximate.

When using a simple, synthetic dataset, where the generative process is controlled, full disentanglement can be verified experimentally (Khemakhem et al., 2020). However, in a realistic setting, the modeling choice for both $q_{\phi}(\mathbf{z}|\mathbf{x},\mathbf{u})$ and $q_{\psi}(\mathbf{z}|\mathbf{u})$ can have an impact on disentanglement. Even when recognition models have enough capacity to fit the data (in our experiments they are Gaussian with diagonal covariance), theoretical guarantees might still fall short, despite the availability of auxiliary variables for all input observations. This could be due to, for example, suboptimal solutions found by the optimization algorithm or to the finite data regime.

5.4.2 Learning an Optimal Conditional Prior

In this work, we advocate for a particular form of a conditional prior, that is the result of learning an optimal representation $\mathbf{z}$, of auxiliary, observed variables $\mathbf{u}$.

In general, an *optimal* representation, for a generic task $\mathbf{y}$ (in our case, we aim at reconstructing $\mathbf{u}$) is defined in terms of sufficiency and minimality: $\mathbf{z}$ is *sufficient* for the task $\mathbf{y}$ if $I(\mathbf{u};\mathbf{y}) = I(\mathbf{z};\mathbf{y})$, where $I(\cdot;\cdot)$ is the mutual information; $\mathbf{z}$ is *minimal* if it compresses the input such that it discards all variability that is not relevant for the task (Achille and Soatto, 2016). As shown in (Tishby et al., 1999), the so called *Information Bottleneck* (IB) can be used to learn an optimal representation $\mathbf{z}$ for the task $\mathbf{y}$, which amounts to optimizing the following Lagrangian:

$$\mathcal{L}_{\text{IB}} = H(\mathbf{y}|\mathbf{u}) + \beta I(\mathbf{u};\mathbf{z}), \tag{5.18}$$

where we denote the entropy by $H(\cdot)$, with the constant β controlling the trade-off between sufficiency and minimality. It is easy to show that Equation 5.18 and Equation 5.15 are equivalent (with $\beta = 1$) when the task is reconstruction (Achille and Soatto, 2016).

In IDVAE, we learn the conditional prior $q_{\psi}(\mathbf{z}|\mathbf{u})$ in part (2) of Equation 5.16, and use it in part (1) by setting $p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u}) = q_{\psi}(\mathbf{z}|\mathbf{u})$. In light of above discussion, this is equivalent to imposing an additional constraint on the conditional prior such that it can learn an *optimal* representation $\mathbf{z}$ from $\mathbf{u}$; the KL term of part (1) pushes $q_{\phi}(\mathbf{z}|\mathbf{x},\mathbf{u})$ toward the optimal conditional prior, which results in superior regularization quality.

Note that Theorem 2 requires auxiliary variables $\mathbf{u}$ to be expressive enough to recover all the independent factors through the parameters $\boldsymbol{\eta}(\mathbf{u})$. In information theoretic terms, $\mathbf{u}$ must be sufficient to recover the ground-truth factors, but there is no explicit need for the extra optimality constraint on $p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{z}|\mathbf{u})$. While Theorem 2 remains valid for an optimal conditional prior, we experimentally demonstrate that, when variational approximations, sub-optimal solutions, or finite data size spoil theoretical results, learning an optimal conditional prior is truly desirable.

5.4.3 A Semi-supervised Variant of IDVAE

So far, we have worked under the assumption that the auxiliary information $\mathbf{u}$ is consistently available for every $\mathbf{x}$. In real scenarios, it is more likely to observe $\mathbf{u}$ for a subset of the input observations. Thus, we propose a variation of IDVAE for a semi-supervised setting. We consider a new objective that consists of two terms (Kingma et al., 2014):

$$\mathcal{L}_{\text{SS-IDVAE}} = \sum_{(\mathbf{x},\mathbf{u})\sim p_l} \mathcal{L}_l(\mathbf{x},\mathbf{u}) + \sum_{\mathbf{x}\sim p_u} \mathcal{L}_u(\mathbf{x}) + \alpha \mathbb{E}_{(\mathbf{x},\mathbf{u})\sim p_l}[\log q_{\boldsymbol{\zeta}}(\mathbf{u}|\mathbf{x})], \tag{5.19}$$

$$\mathcal{L}_l(\mathbf{x},\mathbf{u}) = \mathcal{L}_{\text{IDVAE}}(\mathbf{x},\mathbf{u}), \tag{5.20}$$

$$\mathcal{L}_u(\mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\zeta}}(\mathbf{u}|\mathbf{x})}[\mathcal{L}_l(\mathbf{x},\mathbf{u})] + \mathcal{H}(q_{\boldsymbol{\zeta}}(\mathbf{u}|\mathbf{x})), \tag{5.21}$$

where $\mathcal{L}_l$ and $\mathcal{L}_u$ are the labeled and unlabeled terms respectively; $q_{\boldsymbol{\zeta}}(\mathbf{u}|\mathbf{x})$ in Equation 5.21 is used to derive $\mathbf{u}$ from $\mathbf{x}$ when $\mathbf{u}$ is not provided as input. In Equation 5.19 we include a third term – $\alpha \mathbb{E}_{(\mathbf{x},\mathbf{u})\sim p_l}[\log q_{\boldsymbol{\zeta}}(\mathbf{u}|\mathbf{x})]$, where α controls the relative weight between generative and purely discriminative learning – such that the distribution $q_{\boldsymbol{\zeta}}(\mathbf{u}|\mathbf{x})$ can learn also from labeled data. Clearly, this method also applies to the work from Khemakhem et al. (2020). In our experiments, we set $\alpha = 0.1N$.

5.5 Experiments

5.5.1 Experimental Settings

Methods. We compare IDVAE with three disentanglement methods: β-VAE, FULLVAE, IVAE. β-VAE (Higgins et al., 2017a) is a baseline for indirect matching methods where no ground-truth factor is known at training time and the only way to enforce a disentangled representation is by increasing the strength of the regularization term through the hyper-parameter β. FULLVAE (Locatello et al., 2020b) is the representative of direct matching methods: it can be considered as a standard β-VAE with an additional regularization term, weighted by an hyper-parameter γ, to match the latent space to the target ground-truth factors. As done in the original implementation, we use a binary cross entropy loss for FULLVAE, where the targets are normalized in $[0, 1]$; we also set $\beta = 1$, to measure the impact of the supervised loss term only. IVAE (Khemakhem et al., 2020) is another indirect matching method where the regularization term, weighted again by β, involves a conditional prior. We additionally report the results for the semi-supervised versions of FULLVAE, IVAE, and IDVAE, which we denote as SS-FULLVAE, SS-IVAE[1], SS-IDVAE, respectively. Variational approximations, and the conditional priors,

[1]The original work (Khemakhem et al., 2020) is not semi-supervised. We extended it for our comparative analysis.

Dataset	Size	Ground-truth factors (distinct values)
DSPRITES	737'280	shape(3), scale(6), orientation(40), x(32), y(32)
CARS3D	17'568	elevation(4), azimuth(24), object type (183)
SHAPES3D	480'000	floor color(10), wall color(10), object color(8), object size(8), object type(4), azimuth(15)
SMALLNORB	24'300	category(5), elevation(9), azimuth(18), light(6)

Table 5.1: Main characteristics of the datasets.

are Gaussian distributions with diagonal covariance. All methods have been implemented in PyTorch (Paszke et al., 2019).

Datasets. We consider four common datasets in the disentanglement literature, where observations are images built as a deterministic function of known generative factors: DSPRITES (Higgins et al., 2017a), SHAPES3D (Kim and Mnih, 2018), CARS3D (Reed et al., 2015) and SMALLNORB (LeCun et al., 2004). We have full control on the generative process and explicit access to the ground-truth factors. All ground-truth factors are normalized in the range $[0, 1]$; for discrete factors, we implicitly assume an ordering before applying normalization. All images are reshaped to a 64×64 size. A short description of the datasets is reported in Table 5.1. Implementations of the generative process for each dataset are based on the code provided by Locatello et al. (2019b).

Disentanglement metrics. In the literature, several metrics have been proposed to measure disentanglement, with known advantages and disadvantages, and ability to capture different aspects of disentanglement. We report the results for some of the most popular metrics: beta score (Higgins et al., 2017a), MIG (Chen et al., 2018), SAP (Kumar et al., 2018), modularity and explicitness (Ridgeway and Mozer, 2018), all with values between 0 and 1. The implementation of the metrics is based on Locatello et al. (2019b). We refer the reader to Appendix B.5 for further details.

Experimental protocol. In order to fairly evaluate the impact of the regularization terms, we rely on a solid experimental protocol where all models are trained with the same convolutional architecture, optimizer, hyper-parameters of the optimizer and batch size, that are empirically known to be good for the datasets we tested. The latent dimension $\mathbf{z}$ is fixed to the true number of ground-truth factors. The conditional prior in IVAE is a MLP network; in IDVAE we use a simple MLP VAE. The same architecture is taken for the conditional prior of the semi-supervised counterparts. Moreover, $q_\zeta(\mathbf{u}|\mathbf{x})$ is implemented by a convolutional neural network. Refer to Appendix B.4 for more details.

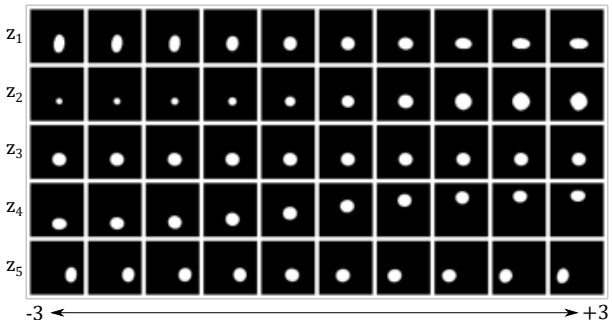


Figure 5.2: Latent traversal of IDVAE model trained on DSPRITES.

We tried six different values of regularization strength associated to the target regularization term of each method – β for β-VAE, IVAE and IDVAE, and γ for FULLVAE: $[1, 2, 4, 6, 8, 16]$. These are recurring values in the disentanglement literature. For each model configuration and dataset, we run the training procedure with 10 random seeds, given that all methods are susceptible to initialization values. After 300'000 training iterations, every model is evaluated according to the disentanglement metrics described above. For FULLVAE, IVAE and IDVAE, all ground-truth factors are provided as input, although IVAE and IDVAE work with a subset of them (or with any other additionally observed variable) as well. We apply the same protocol for the semi-supervised experiments too, where we provide, at training time, all the ground-truth factors for a subset of the input observations only, 1% and 10% respectively. At testing time, $\mathbf{u}$ is instead estimated from $q_\zeta(\mathbf{u}|\mathbf{x})$.

5.5.2 Experimental Results

Qualitative Evaluation. Latent traversal is a simple approach to visualize disentangled representations, by plotting the effects that each latent dimension of a randomly selected sample has on the reconstructed output. In Figure 5.2, we evaluate a configuration (single seed) of our IDVAE model trained on DSPRITES (other datasets in Appendix B.6). Every row of the figure represents a latent dimension we vary in the range $[-3, 3]$, while keeping the other dimensions fixed. We can see that z_1 has learned orientation reasonably well; z_2 is responsible of the object scale; z_4 and z_5 reflect changes on the vertical and horizontal axis, respectively. z_3 tried to learn, without success, shape changes. This means that IDVAE has the capability of learning an *interpretable* disentangled representation, where every dimension of the latent variable $\mathbf{z}$ not only is independent from the others, but also corresponds to a true, semantically meaningful ground-truth factor. Next, we rely on disentanglement metrics to make a quantitative comparison among the tested methods.

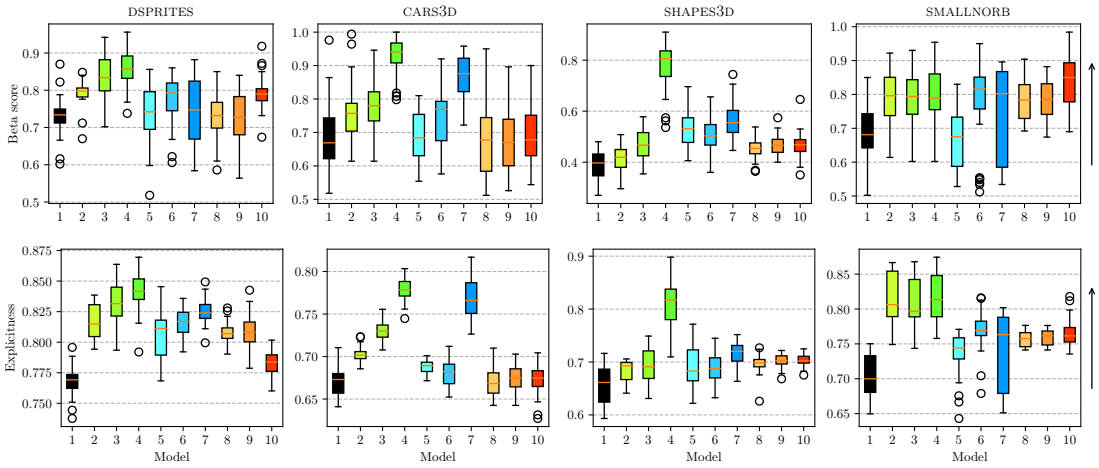


Figure 5.3: Beta score and explicitness (the higher the better). 1=β-VAE, 2=SS-IDVAE (1%), 3=SS-IDVAE (10%), 4=IDVAE, 5=SS-IVAE (1%), 6=SS-IVAE (10%), 7=IVAE, 8=SS-FULLVAE (1%), 9=SS-FULLVAE (10%), 10=FULLVAE. Percentage of labeled samples in parenthesis.

Disentanglement Evaluation. In Figure 5.3, we report, for each method and for each dataset, the ranges of the beta score and explicitness values with a box-plot. The variance of the box-plots is due to the random seeds and regularization strengths, which are the only parameters we vary. Furthermore, Figure 5.3 includes the results for SS-IDVAE, SS-IVAE and SS-FULLVAE (trained with 1% and 10% labeled samples), with different shades of green, blue, and red, respectively. The remaining evaluation metrics can be found in Appendix B.6, but they are essentially all correlated, as also noted in Locatello et al. (2019b).

Overall, we observe, as expected, that β-VAE is often the worst method. Indeed, it has no access to any additional information at training time except the data itself. Despite this, β-VAE disentanglement performance is surprisingly not that far from FULLVAE that directly matches the latent space with the ground-truth factors. In some cases, β-VAE obtains very high beta scores (see outliers), such as for DSPRITES and CARS3D datasets, confirming the sensitivity to random initialization of unsupervised methods (Locatello et al., 2019b). Note also that FULLVAE exhibits inconsistent performance across the four datasets.

IDVAE turns out to be the best method across several disentanglement metrics, except for SMALLNORB, where FULLVAE's beta score is slightly better. For this specific dataset and metric, there are no considerable differences among methods, since most of the box-plots overlap. We note that IDVAE outperforms IVAE: considering that the two methods differ for the way the conditional prior is learned, our experiments show that an

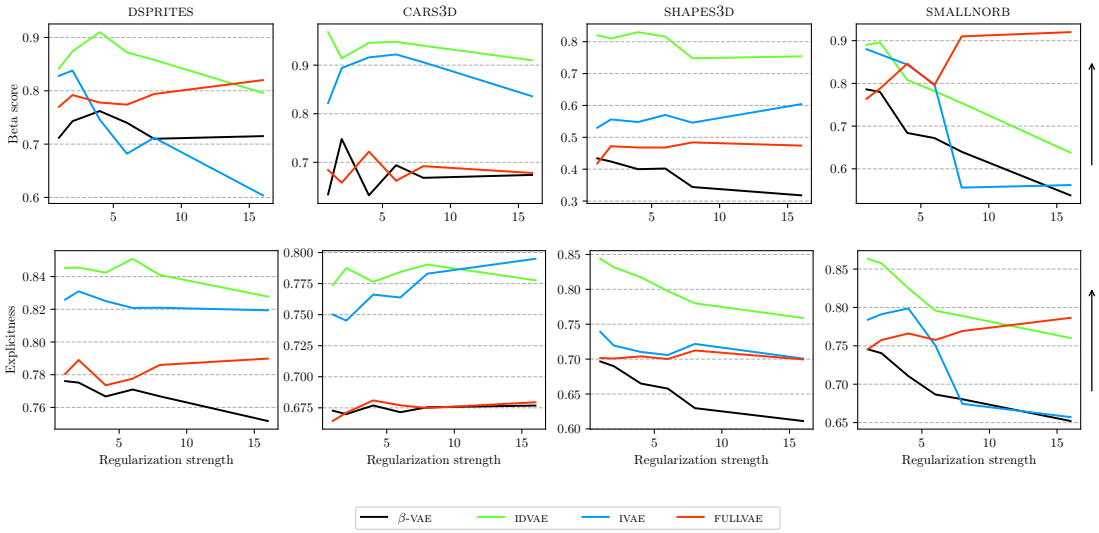


Figure 5.4: Beta score and explicitness median (the higher the better) as a function of the regularization strength.

optimal conditional prior, as we propose in this work, offers substantial benefits in terms of disentanglement and it is the only reason for IDVAE superiority. Finally, although both IVAE and IDVAE have theoretical guarantees on disentanglement and use the full set of ground-truth factors as input, they do not always obtain the maximum evaluation score, in practice. This is in line with the considerations in Section 5.4.1.

The analysis above remains valid if we consider the semi-supervised versions of the tested methods, too. We observe that, with the exception of SMALLNORB, SS-IDVAE's disentanglement performance coherently increases when it observes more labeled instances. The same trend is generally followed by SS-IVAE. SS-FULLVAE, instead, seems to be less susceptible to the number of labeled instances. In general, even a small percentage of labeled instances (1%) is enough for SS-IDVAE to outperform β-VAE and to keep up with FULLVAE that is, however, a fully supervised method. This suggests that SS-IDVAE is a valid choice for applications where collecting additional information about the training data is difficult or expensive.

Impact of the regularization strength. The disentanglement performance of each method might change drastically as a function of the regularization strength: some approaches might work significantly better in some ranges and very badly in others. In Figure 5.4, we plot, for each method and for each dataset, the median of the beta score and explicitness evaluation values as a function of the regularization strength. This is also useful to see if there are methods that consistently dominate others. In this case, we

do not report the results for SS-IDVAE, SS-IVAE, and SS-FULLVAE to make the plots more easily readable. Additional disentanglement score results, including the semi-supervised versions, can be found in Appendix B.6.

Across all the datasets, IDVAE achieves the best median scores for a wide range of regularization strengths. In DSPRITES, CARS3D, and SHAPES3D, IDVAE dominates all the other methods (IDVAE is largely dominant also considering the remaining evaluations metrics). The performance of IVAE and FULLVAE can match that of IDVAE in some datasets, but the behavior is not consistent: if we focus on beta score, IVAE is the second best method in CARS3D and SHAPES3D, whereas in DSPRITES and SMALLNORB, performance drops when we increase the regularization strength – even β-VAE performs better; FULLVAE behaves well for DSPRITES and SMALLNORB, but it is on pair only with β-VAE in CARS3D and SHAPES3D.

By observing the evolution of the disentanglement scores, it appears that there is no clear strategy to choose the regularization strength. For IDVAE, in datasets such as DSPRITES and CARS3D, the regularization strength does not significantly affect the beta score; in SHAPES3D and SMALLNORB, we note instead a decreasing monotonic trend. The situation is similar if we look at the explicitness, but it differs if we consider other disentanglement metrics. It is plausible to deduce that the regularization strength is both model and data specific, and it also depends on the disentanglement metric.

5.5.3 Limitations

In our experimental campaign we use the same convolutional architecture for all the methods we compare. We do not vary the optimization hyper-parameters and the dimension of the latent variables. Hence, we cannot ensure that every method runs in its best conditions. Nevertheless, our experimental protocol is in line with the standards in the literature of disentangled representation learning, makes our analysis independent of method-specific optimizations, and has the benefit of reducing training times.

We use the whole set of ground-truth factors as auxiliary variables, in the semi-supervised settings too, whereas it is possible to study the impact of only a subset of the factors to be available. Moreover, IDVAE and IVAE can use any kind of auxiliary variables, as long as they are informative about the ground-truth factors: they are not restricted to using, e.g., labels corresponding to input data, as we (and several studies) do.

Finally, we do not study the implications and benefits of disentanglement for solving complex downstream tasks, which is an interesting task we leave for future work. It should be noted, however, that most of the disentanglement metrics can be seen as a proxy for the performance of a simple downstream task, i.e., matching each latent dimension with the corresponding ground-truth factor.

5.6 Conclusion

In this work, we went a step further in the design of identifiable generative models to learn disentangled representations. IDVAE uses a prior that encodes ground-truth factor information captured by auxiliary observed variables. The key idea was to learn an optimal representation of the latent space, defined by an inference network on the posterior of the latent variables, given the auxiliary variables. Such posterior is then used as a prior on the latent variables of a second generative model, whose inference network learns a mapping between input observations and latents. We also proposed a semi-supervised version of IDVAE that can be applied when auxiliary variables are available for a subset of the input observations only. Experimental results offer evidence that IDVAE and SS-IDVAE often outperforms existing alternatives to learn disentangled representations, according to several established metrics.

CHAPTER

SIX

CONCLUSIONS

Interpretable machine learning unifies under one umbrella many research fields that, despite being profoundly different, share the same goal: making machine learning understandable and accessible to humans. This is a difficult challenge that we have to win in order to proceed our road toward explainable AI. We conclude this thesis with a summary of the main themes and contributions presented in the previous chapters, with a special focus on future work directions.

6.1 Themes and Contributions

We summarize the main themes and contributions of this work as follows:

Overview of interpretable machine learning. The main challenge when approaching old and continuously evolving fields like interpretable machine learning is to find the way through the huge amount of work, often coming from different research communities, and using inconsistent notations to refer to the same concepts. In the first chapter of the thesis, we gave a unified view of the main approaches to interpretable machine learning, with a shared notation and a common taxonomy. It was a huge effort, considering also that relevant old work did not make explicit reference to "interpretability" or "explainability" in the title. In Chapter 1, aside from the standard categorization between transparent models and post-hoc interpretability techniques, we studied interpretable machine learning from two points of view: interpretability of the models and interpretability of the data representations.

A unified and critical view of rule learning approaches. Regarding the interpretability of machine learning models, predictive rule learning belongs to the family of transparent models and aims at discovering interesting and highly predictive patterns in the data in the form of IF-THEN-ELSE rules. The rule learning process consists of distinct steps, each with its own challenges: feature construction, rule construction, hypothesis construction, and eventually post-pruning. Different implementations of these building blocks result in different methods, that we analyzed in Chapter 2 to highlight their criticalities, mainly related to computational complexity, small disjuncts, predictive performance-interpretability tradeoff.

A novel method to learn interpretable rule sets. In Chapter 3, we proposed an ensemble of bottom-up learners – LIBRE – that is: i) versatile and effective with both balanced and imbalanced data, ii) interpretable, iii) scalable to high-dimensional datasets. The key intuition was to exploit the known advantages of bottom-up learners in imbalanced settings, and improve their generalization capabilities through an interpretable ensembling technique that, in our case, is simply a union. Our experimental campaign shows the superiority of LIBRE, in terms of predictive performance, interpretability and scalability, when compared to state-of-the-art competitors and black-box models.

A unified and critical view of disentanglement methods. Regarding the interpretability of data representations, disentangled representation learning is a branch of representation learning that has the goal of learning representations that are aligned with a series of statistically independent and semantically meaningful ground-truth factors that are assumed to generate the input observations. In Chapter 4, we presented the most recent theoretical frameworks to define and evaluate disentangled representations, reporting applications were disentangled representations ensure practical benefits.

A novel method to learn disentangled representations. In Chapter 5, we followed the formalization of disentanglement presented in the previous chapter to review the state-of-the-art of VAE-based disentanglement methods, highlighting the role of the regularization term and inductive biases on disentanglement. We studied the connection between model identifiability and disentanglement, and proposed IDVAE, a method that exploits a factorized prior, conditionally dependent on auxiliary variables complementing input observations, to improve the regularization quality on the learned representation, and obtain theoretical guarantees on model identifiability, hence disentanglement. IDVAE, and its semi-supervised version, outperforms related state-of-the-art-methods according to several well-established disentanglement metrics.

6.2 Future Work

It is not an exaggeration to say that, at this moment, interpretable machine learning has an edge over any other machine learning field. While I am writing this thesis, there is an uncountable number of remarkable works that would deserve to be discussed. All this to say that interpretable machine learning is a "work-in-progress". In this section, we list interesting directions for future research we believe to be relevant to the theoretical community and practical aspects of interpretable machine learning, with a focus on rule learning and disentangled representation learning. We remind the reader that an extensive discussion about challenges and future work can be found at the end of Chapters 1, 2 and 4.

Interpretable machine learning. The gap between human explanations and AI explanations is significant which is why we should work to reduce it. We suppose that transparent models are the future of interpretable machine learning, and might help to avoid all the issues derived from post-hoc interpretability techniques, as discussed in Section 1.5. We are aware that defining interpretability for specific domains and methods is difficult: for instance, the number and size of rules are standard proxies for interpretability in rule-based methods, but it is not equally clear how to encode and measure interpretability in computer vision methods (although there have been already successful results (Chen et al., 2019)). In this context, also disentangled representation learning is a way to natively incorporate interpretability constraints in the development of generative models.

Rule learning. Rule learning is a mature field with known advantages and historical problems investigated throughout this work that we tried to solve by proposing LIBRE. The optimization problem of discovering rules, eventually including interpretability constraints, is known to be computationally hard. Solving it, or providing approximate solutions, by leveraging new theoretical techniques and advances in hardware is still an open problem. Dealing with class imbalance problems is another fundamental challenge of machine learning that always calls for new approaches. *Incremental rule learning* (Schlimmer and Granger, 1986) and *Bayesian rule learning* (Wang et al., 2017; Yang et al., 2017) are two other interesting future directions that aim at extending rule learning to data streams and probabilistic settings.

Disentangled representation learning. Learning disentangled representations is hard even in the easiest formulation with independent ground-truth factors. Despite non-negligible advancements, recent methods usually do not work well for real datasets. There have been attempts to learn disentangled representations for multiple objects (Greff et al., 2019), hierarchical representations (Singh et al., 2019), and representations

with causal relation among ground-truth factors (Yang et al., 2020). Nevertheless, before considering more complicated formulations, we believe it is crucial to i) obtain consistent results for toy settings first, ii) investigate alternative methods that lead to general theoretical guarantees on disentanglement, not necessarily valid in the limit of infinite data only, iii) design new disentanglement evaluation measures, well aligned with the corresponding theoretical definition of disentanglement and that do not require (a full) apriori knowledge on the generative process or ground-truth factors.

REFERENCES

A. Achille and S. Soatto. Information Dropout: Learning Optimal Representations Through Noise. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40: 2897–2905, 2016.

R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB, 1994.

L. B. Almeida. MISEP - Linear and Nonlinear ICA Based on Mutual Information. *Journal of Machine Learning Research*, 4:1297–1318, 2004.

E. Angelino, N. Larus-Stone, D. Alabi, M. Seltzer, and C. Rudin. Learning Certifiably Optimal Rule Lists for Categorical Data. *Journal of Machine Learning Research*, 18 (234):1–78, 2018.

C. Antaki and I. Leudar. Explaining in Conversation: Towards an Argument Model. *European Journal of Social Psychology*, 22:181–194, 1992.

D. W. Apley and J. Zhu. Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models, 2019.

L. Arras, G. Montavon, K.-R. Müller, and W. Samek. Explaining Recurrent Neural Network Predictions in Sentiment Analysis. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, WASSA, 2017.

A. B. Arrieta, N. D. Rodríguez, J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020.

L. Auret and C. Aldrich. Interpretation of Nonlinear Relationships between Process Variables by use of Random Forests. *Minerals Engineering*, 35:27–42, 2012.

D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller. How to Explain Individual Classification Decisions. *Journal of Machine Learning Research*, 11(61):1803–1831, 2010.

N. Barakat and A. Bradley. Rule Extraction from Support Vector Machines: A Sequential Covering Approach. *Knowledge and Data Engineering, IEEE Transactions on*, 19: 729–741, 2007.

N. Barakat and J. Diederich. Eclectic Rule-Extraction from Support Vector Machines. *International Journal of Computational Intelligence*, 2, 2006.

H. Barlow, T. Kaushal, and G. Mitchison. Finding Minimum Entropy Codes. *Neural Computation*, 1(3):412–423, 1989.

B. Baron and M. Musolesi. Interpretable Machine Learning for Privacy-Preserving Pervasive Systems. *IEEE Pervasive Computing*, 19(1):73–82, 2020.

D. G. T. Barrett, F. Hill, A. Santoro, A. S. Morcos, and T. Lillicrap. Measuring Abstract Reasoning in Neural Networks. In *Proceedings of the 35th International Conference on Machine Learning*, ICML, 2018.

R. J. Bayardo. Brute-Force Mining of High-Confidence Classification Rules. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, KDD, 1997.

Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35: 1798–1828, 2013.

F. Bergadano. Learning Two-Tiered Descriptions of Flexible Concepts: The POSEIDON System. *Machine Learning*, 8:5–43, 1992.

O. Biran and C. V. Cotton. Explanation and Justification in Machine Learning : A Survey. In *Workshop on Explainable Artificial Intelligence*, IJCAI, 2017.

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, 2006.

J. Blaszczynski, M. Deckert, J. Stefanowski, and S. Wilk. Integrating Selective Preprocessing of Imbalanced Data with Ivotes Ensemble. In *Proceedings of the 7th International Conference on Rough Sets and Current Trends in Computing*, RSCTC, 2010.

C. Borgelt. An Implementation of the FP-growth Algorithm. In *Proceedings of the 1st International Workshop on Open Source Data Mining*, OSDM, 2005.

D. Bouchacourt, R. Tomioka, and S. Nowozin. Multi-Level Variational Autoencoder: Learning Disentangled Representations from Grouped Observations. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, AAAI, 2018.

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

L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, 1984.

B. Bringmann, S. Nijssen, and A. Zimmermann. Pattern-Based Classification: A Unifying Perspective. In *Proceedings of the ECML/PKDD-09 Workshop (LeGo-09)*, 2009.

C. Brunk and M. J. Pazzani. An Investigation of Noise-Tolerant Relational Concept Learning Algorithms. In *Proceedings of the 8th International Conference on Machine Learning*, ICML, 1991.

C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding disentangling in β-VAE. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NeurIPS, 2017.

T. Calders, F. Kamiran, and M. Pechenizkiy. Building Classifiers with Independency Constraints. In *2009 IEEE International Conference on Data Mining Workshops*, 2009.

J. Cendrowska. PRISM: An Algorithm for Inducing Modular Rules. *International Journal of Man-Machine Studies*, 27(4):349 – 370, 1987.

A. Chang, D. Bertsimas, and C. Rudin. An Integer Optimization Approach to Associative Classification. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NeurIPS, 2012.

A. C. F. Chaves, M. M. B. R. Vellasco, and R. Tanscheit. Fuzzy Rule Extraction from Support Vector Machines. In *Proceedings of the 5th International Conference on Hybrid Intelligent Systems*, HIS, 2005.

C. Chen and C. Rudin. An Optimization Approach to Learning Falling Rule Lists. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, AISTATS, 2018.

C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su. This Looks Like That: Deep Learning for Interpretable Image Recognition. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS, 2019.

J. Chen and K. Batmanghelich. Weakly Supervised Disentanglement by Pairwise Similarities. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, AAAI, 2020a.

J. Chen and K. Batmanghelich. Robust Ordinal VAE: Employing Noisy Pairwise Comparisons for Disentanglement. *ArXiv*, 2020b.

T. Q. Chen, X. Li, R. B. Grosse, and D. K. Duvenaud. Isolating Sources of Disentanglement in Variational Autoencoders. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2018.

B. Cheung, J. A. Livezey, A. K. Bansal, and B. A. Olshausen. Discovering Hidden Factors of Variation in Deep Networks. In *CoRR*, 2015.

D.-A. Chiang, W. Chen, Y.-F. Wang, and L.-J. Hwang. Rules Generation from the Decision Tree. *Journal of Information Science and Engineering*, 17(2), 2001.

E. Choi, M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart. RETAIN: An Interpretable Predictive Model for Healthcare using Reverse Time Attention Mechanism. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016.

P. Clark and T. Niblett. The CN2 Induction Algorithm. *Machine Learning*, 3(4):261–283, 1989.

P. Clark, P. Clark, T. Niblett, and T. Niblett. Induction in Noisy Domains. In *Proceedings of the 2nd European Working Session on Learning*, EWSL, 1987.

W. W. Cohen. Efficient Pruning Methods for Separate-and-Conquer Rule Learning Systems. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, 1993.

W. W. Cohen. Fast Effective Rule Induction. In *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*, ICML, 1995.

P. Comon. Independent Component Analysis, a New Concept? *Signal Processing*, 36: 287–314, 1994.

C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.

P. Cortez and M. J. Embrechts. Opening black box Data Mining models using Sensitivity Analysis. In *2011 IEEE Symposium on Computational Intelligence and Data Mining*, CIDM, 2011.

P. Cortez and M. J. Embrechts. Using Sensitivity Analysis and Visualization Techniques to Open Black Box Data Mining Models. *Information Sciences*, 225:1–17, 2013.

T. Cover and P. Hart. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

E. Creager, D. Madras, J.-H. Jacobsen, M. Weis, K. Swersky, T. Pitassi, and R. Zemel. Flexibly Fair Representation Learning by Disentanglement. In *Proceedings of the 36th International Conference on Machine Learning*, ICML, 2019.

P. Dabkowski and Y. Gal. Real Time Image Saliency for Black Box Classifiers. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS, 2017.

S. Dash, O. Günlük, and D. Wei. Boolean Decision Rules via Column Generation. In *Proceeding of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2018.

L. De Raedt. *Interactive Theory Revision: An Inductive Logic Programming Approach*. Academic Press Ltd., 1992.

K. Dembczyński, W. Kotłowski, and R. Słowiński. ENDER: A Statistical Framework for Boosting Decision Rules. *Data Mining and Knowledge Discovery*, 21(1):52–90, 2010.

H. Deng. Interpreting Tree Ensembles with inTrees. *International Journal of Data Science and Analytics*, 7(4):277–287, 2019.

J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-scale Hierarchical Image Database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, 2009.

E. L. Denton and v. Birodkar. Unsupervised Learning of Disentangled Representations from Video. In *Proceedings of the 30th International Conference on on Neural Information Processing Systems*, NeurIPS, 2017.

A. Dittadi, F. Träuble, F. Locatello, M. Wüthrich, V. Agrawal, O. Winther, S. Bauer, and B. Schölkopf. On the Transfer of Disentangled Representations in Realistic Settings, 2020.

K. Do and T. Tran. Theory and Evaluation Metrics for Learning Disentangled Representations. In *Proceedings of the 8th International Conference on Learning Representations*, ICLR, 2020.

P. Domingos. Unifying Instance-Based and Rule-Based Induction. *Machine Learning*, 24 (2):141–168, 1996.

P. Domingos. Knowledge Discovery Via Multiple Models. *Intelligent Data Analysis*, 2: 187–202, 1998.

J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell. Long-Term Recurrent Convolutional Networks for Visual Recognition

and Description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):677–691, 2017.

Y. Dong, H. Su, J. Zhu, and B. Zhang. Improving Interpretability of Deep Neural Networks with Semantic Information. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, 2017.

F. Doshi-Velez and B. Kim. Towards A Rigorous Science of Interpretable Machine Learning, 2017.

D. Dua and C. Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

W. Duivesteijn, E. L. Mencía, J. Fürnkranz, and A. J. Knobbe. Multi-label LeGo - Enhancing Multi-label Classifiers with Local Patterns. In *Proceedings of the 11th International Symposium on Advances in Intelligent Data Analysis*, IDA, 2012.

C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through Awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS, 2012.

C. Eastwood and C. K. I. Williams. A Framework for the Quantitative Evaluation of Disentangled Representations. In *Proceedings of the 6th International Conference on Learning Representations*, ICLR, 2018.

B. Esmaeili, H. Wu, S. Jain, A. Bozkurt, N. Siddharth, B. Paige, D. H. Brooks, J. Dy, and J.-W. van de Meent. Structured Disentangled Representations. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, AISTATS, 2020.

D. Fensel and M. Wiese. Refinement of Rule Sets with JoJo. In *Proceedings of the 6th European Conference on Machine Learning*, ECML, 1993.

J. H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics*, 29:1189–1232, 2000.

J. H. Friedman and B. E. Popescu. Predictive Learning via Rule Ensembles. *The Annals of Applied Statistics*, 2(3):916–954, 2008.

X. Fu, C. Ong, S. Keerthi, T. G. G. Hung, and L. Goh. Extracting the Knowledge Embedded in Support Vector Machines. In *IEEE International Conference on Neural Networks - Conference Proceedings*, 2004.

J. Fürnkranz and G. Widmer. Incremental Reduced Error Pruning. In *Proceedings of the 11th International Conference on International Conference on Machine Learning*, ICML, 1994.

J. Fürnkranz, D. Gamberger, and N. Lavrač. *Foundations of Rule Learning*. Springer Publishing Company, Incorporated, 2014.

D. Gamberger and N. Lavrac. Confirmation rule sets. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, KDD, 2000.

D. Gamberger and N. Lavrac. Expert-Guided Subgroup Discovery: Methodology and Application. *Journal of Artificial Intelligence Research*, 17(1), 2002.

B. Gaonkar, R. T. Shinohara, and C. Davatzikos. Interpreting Support Vector Machine Models for Multivariate Group Wise Analysis in Neuroimaging. *Medical Image Analysis*, 24(1):190–204, 2015.

S. T. Goh and C. Rudin. Box Drawings for Learning with Imbalanced Data. In *Proceedings of the 20th International Conference on Knowledge Discovery and Data Mining*, KDD, 2014.

D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., 1989.

A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin. Peeking Inside the Black Box: Visualizing Statistical Learning With Plots of Individual Conditional Expectation. *Journal of Computational and Graphical Statistics*, 24, 2013.

M. W. Gondal, M. Wüthrich, Đorđe Miladinovic, F. Locatello, M. Breidt, V. Volchkov, J. Akpo, O. Bachem, B. Schölkopf, and S. Bauer. On the Transfer of Inductive Bias from Simulation to the Real World: a New Disentanglement Dataset. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS, 2019.

G. Gondim-Ribeiro, P. Tabacof, and E. Valle. Adversarial Attacks on Variational Autoencoders, 2018.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, NeurIPS, 2014.

R. Goroshin, M. F. Mathieu, and Y. LeCun. Learning to Linearize Under Uncertainty. In *Proceedings of the 28th International Conference on on Neural Information Processing Systems*, NeurIPS, 2015.

M. Greaves, H. Holmback, and J. Bradshaw. What Is a Conversation Policy? *Lecture Notes in Computer Science*, 1916, 1999.

K. Greff, R. L. Kaufman, R. Kabra, N. Watters, C. Burgess, D. Zoran, L. Matthey,

M. Botvinick, and A. Lerchner. Multi-Object Representation Learning with Iterative Variational Inference. In *Proceedings of the 36th International Conference on Machine Learning*, ICML, 2019.

J. W. Grzymala-busse and J. Stefanowski. Three discretization methods for rule induction. *International Journal of Intelligent Systems*, 16(1):29–38, 2001.

J. W. Grzymala-Busse, L. K. Goodwin, W. J. Grzymala-Busse, and X. Zheng. An Approach to Imbalanced Data Sets Based on Changing Rule Strength. In *Rough-Neural Computing: Techniques for Computing with Words*, 2000.

L. Gu, J. Li, H. He, G. Williams, S. Hawkins, and C. Kelman. Association Rule Discovery with Unbalanced Class Distributions. In *Proceedings of the 16th Australian Conference on Artificial Intelligence*, AI, 2003.

D. Gunning. DARPA's Explainable Artificial Intelligence (XAI) Program. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*, IUI. Association for Computing Machinery, 2019.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18, 2009.

S. Hara and K. Hayashi. Making Tree Ensembles Interpretable: A Bayesian Model Selection Approach. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, AISTATS, 2018.

M. Hardt, E. Price, and N. Srebro. Equality of Opportunity in Supervised Learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NeurIPS, 2016.

S. Harmeling, A. Ziehe, M. Kawanabe, and K. Müller. Kernel-Based Nonlinear Blind Source Separation. *Neural Computation*, 15:1089–1124, 2003.

T. Hastie and R. Tibshirani. *Generalized Additive Models*. Wiley Online Library, 1990.

J. R. Hauser, O. Toubia, T. Evgeniou, R. Befurt, and D. Dzyabura. Disjunctions of Conjunctions, Cognitive Simplicity, and Consideration Sets. *Journal of Marketing Research*, 47(3):485–496, 2010.

D. M. Hawkins. The Problem of Overfitting. *Journal of Chemical Information and Computer Sciences*, 44(1):1–12, 2004.

F. Heider and M. Simmel. An Experimental Study of Apparent Behavior. *American Journal of Psychology*, 57:243–259, 1944.

L. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell. Generating

Visual Explanations. In *Proceedings of the 14th Europena Conference on Computer Vision*, ECCV, 2016.

B. Herman. The Promise and Peril of Human Evaluation for Model Interpretability. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2017.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR, 2017a.

I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner. DARLA: Improving Zero-Shot Transfer in Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*, ICML, 2017b.

I. Higgins, D. Amos, D. Pfau, S. Racaniere, L. Matthey, D. Rezende, and A. Lerchner. Towards a Definition of Disentangled Representations, 2018a.

I. Higgins, N. Sonnerat, L. Matthey, A. Pal, C. P. Burgess, M. Bošnjak, M. Shanahan, M. Botvinick, D. Hassabis, and A. Lerchner. SCAN: Learning Hierarchical Compositional Visual Concepts. In *Proceedings of the 6th International Conference on Learning Representations*, ICLR, 2018b.

D. J. Hilton. Conversational Processes and Causal Explanation. *Psychological Bulletin*, 107:65–81, 1990.

D. J. Hilton. Mental Models and Causal Explanation: Judgements of Probable Cause and Explanatory Relevance. *Thinking and Reasoning*, 2(4):273–308, 1996.

G. E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming Auto-Encoders. In *Proceedings of the 21th International Conference on Artifificial Neural Networks*, ICANN, 2011.

T. K. Ho. The Random Subspace Method for Constructing Decision Forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

M. D. Hoffman and M. J. Johnson. Elbo Surgery: Yet Another Way to Carve Up the Variational Evidence Lower Bound. In *Proceedings of the 29th International Conference on Neural Information Processing Systems*, NeurIPS, 2016.

R. C. Holte, L. E. Acker, and B. W. Porter. Concept Learning and the Problem of Small Disjuncts. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, IJCAII, 1989.

H. Hosoya. Group-based Learning of Disentangled Representations with Generalizability for Novel Contents. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI, 2019.

J.-T. Hsieh, B. Liu, D.-A. Huang, L. F. Fei-Fei, and J. C. Niebles. Learning to Decompose and Disentangle Representations for Video Prediction. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2018.

A. Hyvärinen and H. Morioka. Unsupervised Feature Extraction by Time-Contrastive Learning and Nonlinear ICA. In *Proceedings of the 29th International Conference on Neural Information Processing Systems*, NeurIPS, 2016.

A. Hyvärinen and H. Morioka. Nonlinear ICA of Temporally Dependent Stationary Sources. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, AISTATS, 2017.

A. Hyvärinen and E. Oja. Independent Component Analysis: Algorithms and Applications. *Neural networks*, 13:411–430, 2000.

A. Hyvärinen and P. Pajunen. Nonlinear Independent Component Analysis: Existence and Uniqueness Results. *Neural networks*, 12:429–439, 1999.

A. Hyvärinen, H. Sasaki, and R. E. Turner. Nonlinear ICA Using Auxiliary Variables and Generalized Contrastive Learning. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, AISTATS, 2019.

P. Jackson. *Introduction to Expert Systems*. Addison-Wesley Longman Publishing Co., Inc., 1998.

F. Janssen and J. Fürnkranz. A Re-Evaluation of the Over-Searching Phenomenon in Inductive Rule Learning. In *Proceedings of the SIAM International Conference on Data Mining*, SIAM, 2009.

A. Jarovsky, T. Milo, S. Novgorodov, and W.-C. Tan. GOLDRUSH: Rule Sharing System for Fraud Detection. *Proceedings of the VLDB Endowment*, 11(12):1998–2001, 2018.

T. Jo and N. Japkowicz. Class Imbalances versus Small Disjuncts. *ACM SIGKDD Explorations Newsletters*, 6(1):40–49, 2004.

V. Jovanoski and N. Lavrac. Classification Rule Learning with APRIORI-C. In *Proceedings of the 10th Portuguese Conference on Artificial Intelligence on Progress in Artificial Intelligence, Knowledge Extraction, Multi-Agent Systems, Logic Programming and Constraint Solving*, EPIA, 2001.

A. Karpathy, J. Johnson, and L. Fei-Fei. Visualizing and Understanding Recurrent Networks, 2015.

R. Kerber. ChiMerge: Discretization of Numeric Attributes. In *Proceedings of the 10th National Conference on Artificial Intelligence*, AAAI, 1992.

I. Khemakhem, D. P. Kingma., R. P. Mont, and A. Hyvärinen. Variational Autoencoders and Nonlinear ICA: A Unifying Framework. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, AISTATS, 2020.

N. Kilbertus, M. Rojas-Carulla, G. Parascandolo, M. Hardt, D. Janzing, and B. Schölkopf. Avoiding Discrimination through Causal Reasoning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2017.

H. Kim and A. Mnih. Disentangling by Factorising. In *Proceedings of the 35th International Conference on Machine Learning*, ICML, 2018.

D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations*, ICLR, 2014.

D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised Learning with Deep Generative Models. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, NeurIPS, 2014.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598), 1983.

A. R. Klivans and R. A. Servedio. Toward Attribute Efficient Learning of Decision Lists and Parities. *Journal of Machine Learning Research*, 7:587–602, 2006.

J. Klys, J. Snell, and R. Zemel. Learning Latent Subspaces in Variational Autoencoders. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2018.

W. Klösgen. Explora: A Multipattern and Multistrategy Discovery Assistant. In *Advances in Knowledge Discovery and Data Mining*, KDD, 1996.

A. J. Knobbe, B. Crémilleux, J. Fürnkranz, and M. Scholz. From Local Patterns to Global Models: The LeGo Approach to Data Mining. In *Proceedings of the ECML/PKDD-08 Workshop (LeGo-08)*, 2008.

P. W. Koh and P. Liang. Understanding Black-Box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning - Volume*, ICML, 2017.

R. Konig, U. Johansson, and L. Niklasson. G-REX: A Versatile Framework for Evolutionary Data Mining. In *Proceedings of the 2008 IEEE International Conference on Data Mining Workshops*, 2008.

J. Kos, I. Fischer, and D. Song. Adversarial Examples for Generative Models. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 36–42, 2018.

B. Kovalerchuk, E. Triantaphyllou, A. S. Deshpande, and E. Vityaev. Interactive Learning of Monotone Boolean Functions. *Information Sciences*, 94:87–118, 1996.

V. Krakovna and F. Doshi-Velez. Increasing the Interpretability of Recurrent Neural Networks Using Hidden Markov Models, 2016.

T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum. Deep Convolutional Inverse Graphics Network. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NeurIPS, 2015.

A. Kumar and B. Poole. On Implicit Regularization in β-VAEs. In *Proceedings of the 37th International Conference on Machine Learning*, ICML, 2020.

A. Kumar, P. Sattigeri, and A. Balakrishnan. Variational Inference of Disentangled Latent Concepts from Unlabeled Observations. In *Proceedings of the 6th International Conference on Learning Representations*, ICLR, 2018.

H. Lakkaraju, S. H. Bach, and J. Leskovec. Interpretable Decision Sets: A Joint Framework for Description and Prediction. In *Proceedings of the 22nd International Conference on Knowledge Discovery and Data Mining*, KDD, 2016.

H. Lakkaraju, N. Arsov, and O. Bastani. Robust and Stable Black Box Explanations. In *Proceedings of the 37th International Conference on Machine Learning*, ICML, 2020.

P. Langley, B. Meadows, M. Sridharan, and D. Choi. Explainable Agency for Intelligent Autonomous Systems. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, AAAI'17, 2017.

A. Laversanne-Finot, A. Pere, and P.-Y. Oudeyer. Curiosity Driven Exploration of Learned Disentangled Goal Spaces. In *Proceedings of the 2nd Conference on Robot Learning*, ICRL, 2018.

N. Lavrac and S. Dzeroski. *Inductive Logic Programming: Techniques and Applications*. Routledge, 1993.

N. Lavrač, S. Džeroski, and M. Grobelnik. Learning Nonrecursive Definitions of Relations with Linus. In *Proceedings of the 5th European Conference on European Working Session on Learning*, EWSL, 1991.

Y. LeCun, F. J. Huang, and L. Bottou. Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR, 2004.

C. Leondes. *Expert Systems: The Technology of Knowledge Management and Decision Making for the 21st Century.* Academic Press, 2002.

D. K. Lewis. *Causal Explanation, Philosophical Papers 2.* Oxford Scholarship Online, 1986.

W. Li, J. Han, and J. Pei. CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, ICDM, 2001.

Y. Li and S. Mandt. Disentangled Sequential Autoencoder. In *Proceedings of the 35th International Conference on Machine Learning*, ICML, 2018.

Z. C. Lipton. The Mythos of Model Interpretability. *Communications of the ACM*, 61 (10):36–43, 2018.

B. Liu, W. Hsu, and Y. Ma. Integrating Classification and Association Rule Mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, KDD, 1998.

B. Liu, Y. Ma, and C. K. Wong. Improving an Association Rule Based Classifier. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery in Databases"*, PKDD, 2000.

F. Locatello, G. Abbati, T. Rainforth, S. Bauer, B. Schölkopf, and O. Bachem. On the Fairness of Disentangled Representations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS, 2019a.

F. Locatello, S. Bauer, M. Lucic, S. Gelly, B. Schölkopf, and O. Bachem. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. In *Proceedings of the 36th International Conference on Machine Learning*, ICML, 2019b.

F. Locatello, B. Poole, G. Rätsch, B. Schölkopf, O. Bachem, and M. Tschannen. Weakly-Supervised Disentanglement Without Compromises. In *Proceedings of the 37th International Conference on Machine Learning*, ICML, 2020a.

F. Locatello, M. Tschannen, S. Bauer, G. Rätsch, B. Schölkopf, and O. Bachem. Disentangling Factors of Variation Using Few Labels. In *Proceedings of the 8th International Conference on Learning Representations*, ICLR, 2020b.

T. Lombrozo. The Structure and Function of Explanations. *Trends in Cognitive Sciences*, 10(10):464,470, 2006.

R. Lopez, J. Regier, M. I. Jordan, and N. Yosef. Information Constraints on Auto-

Encoding Variational Bayes. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2018.

J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under Concept Drift: A Review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363, 2019.

S. M. Lundberg and S.-I. Lee. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2017.

A. Mahendran and A. Vedaldi. Understanding Deep Image Representations by Inverting Them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, 2015.

A. Makhzani and B. J. Frey. PixelGAN Autoencoders. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NeurIPS, 2017.

D. Malioutov and K. Varshney. Exact Rule Learning via Boolean Compressed Sensing. In *Proceedings of the 30th International Conference on Machine Learning*, ICML, 2013.

B. Malle. *How the Mind Explains Behavior: Folk Explanations, Meaning, and Social Interaction*. The MIT Press, 2004.

E. Mathieu, T. Rainforth, N. Siddharth, and Y. W. Teh. Disentangling Disentanglement in Variational Autoencoders. In *Proceedings of the 36th International Conference on Machine Learning*, ICML, 2019.

P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman and Hall/CRC, 1989.

R. S. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The Multi-Purpose Incremental Learning System AQ15 and Its Testing Application to Three Medical Domains. In *Proceedings of the 5th AAAI National Conference on Artificial Intelligence*, AAAI, 1986.

T. Miller. Explanation in Artificial Intelligence: Insights from the Social Sciences. *Artificial Intelligence*, 267:1–38, 2019.

T. Milo, S. Novgorodov, and W.-C. Tan. Interactive Rule Refinement for Fraud Detection. In *Proceedings of the 21st International Conference on Extending Database Technology*, EDBT, 2018.

T. M. Mitchell. Generalization as Search. *Artificial Intelligence*, 18(2):203–226, 1982.

D. Mladenic. Combinatorial Optimization in Inductive Concept Learning. In *Proceedings*

of the 10th International Conference on International Conference on Machine Learning, ICML, 1993.

G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. Explaining Nonlinear Classification Decisions with Deep Taylor Decomposition. *Pattern Recognition*, 65(C):211–222, 2017.

S. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13, 1995.

S. Muggleton and C. Feng. Efficient Induction of Logic Programs. In *New Generation Computing*, 1990.

M. Muselli. Approximation Properties of Positive Boolean Functions. In *Proceedings of the 16th Italian Workshop on Neural Nets and and International Workshop on Natural and Artificial Immune Systems*, WIRN/NAIS, 2005.

M. Muselli. Switching Neural Networks: A New Connectionist Model for Classification. In *Neural Nets*, 2006.

M. Muselli and A. Quarati. Shadow Clustering : A Method for Monotone Boolean Function Synthesis, 2004.

M. Muselli and A. Quarati. Reconstructing positive Boolean functions with shadow clustering. In *Proceedings of the 2005 European Conference on Circuit Theory and Design*, ECCTD, 2005.

K. Napierala. BRACID: A Comprehensive Approach to Learning Rules from Imbalanced Data. *Journal of Intelligent Information Systems*, 39(2):335–373, 2012.

A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune. Synthesizing the Preferred Inputs for Neurons in Neural Networks via Deep Generator Networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NeurIPS, 2016.

C. H. Nguyen and T. Ho. An Imbalanced Data Rule Learner. In *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD, 2005.

T. Niblett and I. Bratko. Learning Decision Rules in Noisy Domains. In *Proceedings of the 6Th Annual Technical Conference on Research and Development in Expert Systems III*, 1987.

H. Nuñez, C. Angulo, and A. Català. Rule Extraction from Support Vector Machines. In *Proceedings of the European Symposium on Artificial Neural Networks*, ESANN, 2002.

H. Nuñez, C. Angulo, and A. Català. Rule-Based Learning Systems for Support Vector Machines. *Neural Processing Letters*, 24(1):1–18, 2006.

F. Offert. "I know it when I see it". Visualization and Intuitive Interpretability, 2017.

C. Olah, A. Mordvintsev, and L. Schubert. Feature Visualization. *Distill*, 2017. doi: 10.23915/distill.00007. https://distill.pub/2017/feature-visualization.

G. Pagallo and D. Haussler. Boolean Feature Discovery in Empirical Learning. *Machine Learning*, 5(1):71–99, 1990.

N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering Frequent Closed Itemsets for Association Rules. In *Proceedings of the 7th International Conference on Database Theory*, ICDT, 1999.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS, 2019.

J. Pearl. *Causality*. Cambridge University Press, 2009.

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

J. Peters, D. Janzing, and B. Schlkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. The MIT Press, 2017.

U. Pompe, M. Kovacic, and I. Kononenko. SFOIL: Stochastic Approach to Inductive Logic Programming. In *Proceedings of the 4th International Workshop on Inductive Logic Programming*, ILP, 1993.

J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1:81–106, 1986.

J. R. Quinlan. Generating Production Rules from Decision Trees. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, IJCAI, 1987a.

J. R. Quinlan. Simplifying Decision Trees. *International Journal of Man-Machine Studies*, 27(3):221–234, 1987b.

J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., 1993.

J. R. Quinlan and R. M. Cameron-Jones. FOIL: A Midterm Report. In *Proceedings of the 4th European Conference on Machine Learning*, ECML, 1993.

J. R. Quinlan and R. M. Cameron-Jones. Oversearching and Layered Search in Empirical Learning. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, IJCAI, 1995.

S. Reed, K. Sohn, Y. Zhang, and H. Lee. Learning to Disentangle Factors of Variation with Manifold Interaction. In *Proceedings of the 31st International Conference on Machine Learning*, ICML, 2014.

S. E. Reed, Y. Zhang, Y. Zhang, and H. Lee. Deep Visual Analogy-Making. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NeurIPS. Curran Associates, Inc., 2015.

D. J. Rezende, S. Mohamed, and D. Wiestra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning*, ICML, 2014.

M. T. Ribeiro, S. Singh, and C. Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD, 2016.

K. Ridgeway and M. C. Mozer. Learning Deep Disentangled Embeddings With the F-Statistic Loss. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS, 2018.

P. R. Rijnbeek and J. A. Kors. Finding a Short and Accurate Decision Rule in Disjunctive Normal Form by Exhaustive Search. *Machine Learning*, 80(1):33–62, 2010.

R. L. Rivest. Learning Decision Lists. *Machine Learning*, 2(3):229–246, 1987.

M. Robnik-Šikonja and I. Kononenko. Explaining Classifications For Individual Instances. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):589–600, 2008.

L. Rosenbaum, G. Hinselmann, A. Jahn, and A. Zell. Interpreting Linear Support Vector Machine Models with Heat Map Molecule Coloring. *Journal of Cheminformatics*, 3:11, 2011.

U. Rückert and S. Kramer. Stochastic Local Search in K-Term DNF Learning. In *Proceedings of the 20th International Conference on International Conference on Machine Learning*, ICML, 2003.

U. Rückert and L. D. Raedt. An Experimental Evaluation of Simplicity in Rule Learning. *Artificial Intelligence*, 172(1):19–28, 2008.

C. Rudin. Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. *Nature Machine Intelligence*, 1:206–215, 2019.

W. Salmon. *Four Decades of Scientific Explanation.* University of Minnesota Press, 1989.

S. Salzberg. A Nearest Hyperrectangle Learning Method. *Machine Learning*, 6(3): 251–276, 1991.

M. Sato and H. Tsukimoto. Rule Extraction from Neural Networks via Decision Tree Induction. In *Proceedings of the International Joint Conference on Neural Network*, IJCNN, 2001.

J. C. Schlimmer and R. H. Granger. Incremental Learning from Noisy Data. *Machine Learning*, 1(3):317–354, 1986.

A. Shrikumar, P. Greenside, and A. Kundaje. Learning Important Features through Propagating Activation Differences. In *Proceedings of the 34th International Conference on Machine Learning*, ICML, 2017.

R. Shu, Y. Chen, A. Kumar, S. Ermon, and B. Poole. Weakly Supervised Disentanglement with Guarantees. In *Proceedings of the 8th International Conference on Learning Representations*, ICLR, 2020.

N. Siddharth, B. Paige, J. van de Meent, A. Desmaison, N. Goodman, P. Kohli, F. Wood, and P. Torr. Learning Disentangled Representations with Semi-Supervised Deep Generative Models. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NeurIPS, 2017.

K. Simonyan, A. Vedaldi, and A. Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In *Proceedings of the 2nd International Conference on Learning Representations*, ICLR, 2014.

K. K. Singh, U. Ojha, and Y. J. Lee. FineGAN: Unsupervised Hierarchical Disentanglement for Fine-Grained Object Generation and Discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, 2019.

P. Smyth and R. M. Goodman. An Information Theoretic Approach to Rule Induction from Databases. *IEEE Transactions on Knowledge and Data Engineering*, 4(4):301–316, 1992.

H. Sprekeler, T. Zito, and L. Wiskott. An Extension of Slow Feature Analysis for Nonlinear Blind Source Separation. *Journal of Machine Learning Research*, 15:921–947, 2014.

X. Steenbrugge, S. Leroux, T. Verbelen, and B. Dhoedt. Improving Generalization for Abstract Reasoning Tasks Using Disentangled Feature Representations. In *Workshop on Relational Representation Learning*, NeurIPS, 2018.

E. Strumbelj and I. Kononenko. An Efficient Explanation of Individual Classifications Using Game Theory. *Journal of Machine Learning Research*, 11:1–18, 2010.

G. Su, D. Wei, K. R. Varshney, and D. M. Malioutov. Learning Sparse Two-Level

Boolean Rules. In *Proceedings of the IEEE 26th International Workshop on Machine Learning for Signal Processing*, MLSP, 2016.

C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *Proceedubgs of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, 2015.

P. Tabacof, J. Tavares, and E. Valle. Adversarial Images for Variational Autoencoders. In *Workshop on Adversarial Training*, NeurIPS, 2016.

S. Tan, R. Caruana, G. Hooker, and Y. Lou. Distill-and-Compare: Auditing Black-Box Models Using Transparent Model Distillation. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, AIES, 2018.

Y. Tan, J. Wang, and J. M. Zurada. Nonlinear Blind Source Separation Using a Radial Basis Function Network. *IEEE Transactions on Neural Networks*, 12:124–134, 2001.

J. B. Tenenbaum and W. T. Freeman. Separating Style and Content with Bilinear Models. *Neural Computation*, 12:1247–1283, 2000.

P. Thagard. Explanatory Coherence. *Behavioral and Brain Sciences*, 12(3):435–467, 1989.

J. J. Thiagarajan, B. Kailkhura, P. Sattigeri, and K. N. Ramamurthy. TreeView: Peeking into Deep Neural Networks Via Feature-Space Partitioning. In *Proceedings of the NIPS 2016 Workshop on Interpretable Machine Learning in Complex Systems*, NeurIPS, 2016.

R. Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society*, 58:267–288, 1996.

N. Tishby, F. C. Pereira, and W. Bialek. The Information Bottleneck Method. In *Proceedings of the 34th Annual Allerton Conference on Communication, Control and Computing*, 1999.

M. Titsias and M. Lázaro-Gredilla. Doubly Stochastic Variational Bayes for non-Conjugate Inference. In *Proceedings of the 31st International Conference on Machine Learning*, ICML, 2014.

G. Tolomei, F. Silvestri, A. Haines, and M. Lalmas. Interpretable Predictions of Tree-Based Ensembles via Actionable Feature Tweaking. In *Proceedings of the 23rd International Conference on Knowledge Discovery and Data Mining*, KDD, 2017.

V. I. Torvik and E. Triantaphyllou. Minimizing the Average Query Complexity of

Learning Monotone Boolean Functions. *INFORMS Journal on Computing*, 14:144–174, 2002.

M. Tschannen, O. F. Bachem, and M. Lučić. Recent Advances in Autoencoder-Based Representation Learning. In *3rd Workshop on Bayesian Deep Learning Workshop*, NeurIPS, 2018.

J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano. Experimental Perspectives on Learning from Imbalanced Data. In *Proceedings of the 24th International Conference on Machine Learning*, ICML, 2007.

S. van Steenkiste, F. Locatello, J. Schmidhuber, and O. Bachem. Are Disentangled Representations Helpful for Abstract Visual Reasoning? In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS, 2019.

N. Vasilyeva, D. A. Wilkenfeld, and T. Lombrozo. Goals Affect the Perceived Quality of Explanations. In D. C. Noelle, R. Dale, A. S. Warlaumont, J. Yoshimi, T. Matlock, C. D. Jennings, and P. P. Maglio, editors, *Proceedings of the 37th Annual Meeting of the Cognitive Science Society*, COGSCI, 2015.

R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing Motion and Content for Natural Video Sequence Prediction. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR, 2017.

W. W. Cohen and Y. Singer. A Simple, Fast, and Effective Rule Learner. In *Proceedings of the 16th National Conference on Artificial Intelligence*, AAAI, 1999.

F. Wang and C. Rudin. Falling Rule Lists. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, AISTATS, 2015.

T. Wang, C. Rudin, F. Doshi-Velez, Y. Liu, E. Klampfl, and P. MacNeille. Or's of And's for Interpretable Classification, with Application to Context-Aware Recommender Systems. In *CoRR in arXiv*, 2015.

T. Wang, C. Rudin, F. Doshi-Velez, Y. Liu, E. Klampfl, and P. MacNeille. A Bayesian Framework for Learning Rule Sets for Interpretable Classification. *Journal of Machine Learning Research*, 18(70):1–37, 2017.

G. I. Webb. Systematic Search for Categorical Attribute-Value Data-Driven Machine Learning. In *Proceedings of the 6th Australian Joint Conference of Artificial Intelligence*, AI, 1993.

G. I. Webb. OPUS: An Efficient Admissible Algorithm for Unordered Search. *Journal of Artificial Intelligence Research*, 5, 1995.

I. Wegener. *The Complexity of Boolean Functions*, page 5. John Wiley and Sons, Inc., 1987.

G. M. Weiss. Learning with Rare Cases and Small Disjuncts. In *In Proceedings of 12th International Conference on Machine Learning*, ICML, 1995.

G. M. Weiss. Mining with Rarity: A Unifying Framework. *SIGKDD Explorations*, 6(1): 7–19, 2004.

G. M. Weiss and H. Hirsh. A Quantitative Study of Small Disjuncts. In *Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence*, AAAI, 2000.

S. M. Weiss and N. Indurkhya. Lightweight Rule Induction. In *Proceedings of the 17th International Conference on Machine Learning*, ICML, 2000.

G. Widmer and M. Kubat. Learning in the Presence of Concept Drift and Hidden Contexts. *Machine Learning*, 23(1):69–101, 1996.

M. Willetts, A. Camuto, T. Rainforth, S. Roberts, and C. Holmes. Improving VAEs' Robustness to Adversarial Attack, 2020.

J. Woodward. *Making Things Happen: A Theory of Causal Explanation.* Oxford University Press, 2004.

S. Wrobel. An Algorithm for Multi-Relational Discovery of Subgroups. In *Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery*, PKDD, 1997.

T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang. The Application of Two-Level Attention Models in Deep Convolutional Neural Network for Fine-grained Image Classification. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, 2015.

K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML, 2015.

H. Yang, C. Rudin, and M. Seltzer. Scalable Bayesian Rule Lists. In *Proceedings of the 34th International Conference on Machine Learning*, ICML, 2017.

J. Yang, S. Reed, M.-H. Yang, and H. Lee. Weakly-Supervised Disentangling with Recurrent Transformations for 3D View Synthesis. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NeurIPS, 2015.

M. Yang, F. Liu, Z. Chen, X. Shen, J. Hao, and J. Wang. CausalVAE: Disentangled Representation Learning via Neural Structural Causal Models, 2020.

X. Yin and J. Han. CPAR: Classification Based on Predictive Association Rules. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, SDM, 2003.

M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica. Apache Spark: A Unified Engine for Big Data Processing. *Communications of the ACM*, 59(11):56–65, 2016.

M. D. Zeiler, G. W. Taylor, and R. Fergus. Adaptive Deconvolutional Networks for Mid and High Level Feature Learning. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV, 2011.

S. Zhao, J. Song, and S. Ermon. InfoVAE: Balancing Learning and Inference in Variational Autoencoders. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, AAAI, 2019.

S. Zhou, E. Zelikman, F. Lu, A. Y. Ng, G. Carlsson, and S. Ermon. Evaluating the Disentanglement of Deep Generative Models through Manifold Topology, 2020.

Z. Zhu, P. Luo, X. Wang, and X. Tang. Multi-View Perceptron: A Deep Model for Learning Face Identity and View Representations. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, NeurIPS, 2014.

J. Zilke, E. Mencía, and F. Janssen. DeepRED – Rule Extraction from Deep Neural Networks. In *Proceeding of the 19th International Conference on Discovery Science*, ICDS, 2016.

B. Üstün, W. Melssen, and L. Buydens. Visualisation and Interpretation of Support Vector Regression models. *Analytica Chimica Acta*, 595(1):299–309, 2007.

APPENDIX

A

ADDITIONAL DETAILS AND RESULTS ABOUT LIBRE

A.1 The Base Method Step-by-step

In this section, we detail the main steps of the base algorithm, by using a concrete example. Consider the scenario of forecasting the failure condition of an IT system from two values representing the CPU and main memory (MEM) utilization, as depicted in the first two columns of Table A.1. We assume that CPU and MEM are continuous features with values in the domain $[0, 100]$. The state of the system is described by a binary *Label*, where 1 represents a system failure. The example reports eight records, two of which are failures.

ID	CPU	MEM	r_1	r_2	String	Label
χ_1	95	10	3	1	110 01	1
χ_2	80	10	1	1	011 01	0
χ_3	81	85	2	2	101 10	1
χ_4	10	85	1	2	011 10	0
χ_5	10	10	1	1	011 01	0
χ_6	82	10	2	1	101 01	0
χ_7	85	10	2	1	101 01	0
χ_8	81	10	2	1	101 01	0

Table A.1: Original values from CPU and MEM, their mappings to discrete ranges (r_1, r_2), binary encoding, and binary label.

Discretization and binarization. The first operation to do is discretization. Assume the discretization algorithm identifies three intervals for CPU and two intervals for MEM,

as follows. CPU: $[0, 81), [81, 95), [95, max]$. MEM: $[0, 85), [85, max]$. We can now map the original values to integer values over the ranges (1, 2, 3) and (1, 2), as shown in columns r_1, r_2, respectively. The resulting discretized records are then mapped to (inverse one-hot encoded) binary strings of five bits, as recorded in the *String* column. We also define a partial order relation between binary records, such that $\mathbf{x} \leq \mathbf{x}' \iff \mathbf{x} \bigwedge \mathbf{x}' = \mathbf{x}'$. Moreover, the application of inverse one-hot encoding ensures that the relation between input features and labels is monotone, according to Definition 3.3.2 in Chapter 3. We can give you an intuition through a simple example: consider two binary strings 011 and 110; we see that $011 \not\leq 110$ and $110 \not\leq 011$, so the relation always holds, independently from the label.

Learning the boundary. Consider the first positive sample χ_1 with string 110 01. An exhaustive search strategy would explore all possible flipping alternatives for the most general conflict-free binary strings. If, for example, we flip-off the first bit we obtain 010 01 $<= \chi_2$: hence, we have a conflict. If, for example, we keep the first bit at 1 and flip-off the second bit, we obtain 100 01, which is in conflict with $\chi_6 - \chi_8$. Finally, if we flip-off the last bit, we obtain 110 00, which has no conflict: this is a candidate boundary point. If we repeat the same procedure for χ_3, after flipping-off the third bit, we get another boundary point 100 10.

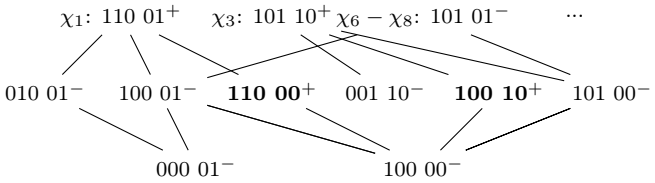


Figure A.1: Partially ordered set created from the records in Table A.1.

Figure A.1 shows the partially ordered set corresponding to Table A.1. At the beginning, the nodes at the top are the ones for which we know the label represented with a superscript symbol + and − for positive and negative, respectively. They can be seen as maximally-specific rules. If we take as target the positive class, we move inside the Boolean lattice by flipping-off positive bits, starting from the positive binary samples, and go down to find binary elements – located on the boundary – that divide positive and negative samples. While we navigate the Boolean lattice, nodes are labelled according to the cover test against the negative samples. As soon as a conflict is found, we can avoid going down from that node, but there is still the possibility to explore that path from another binary sample. This recursive procedure corresponds to up-and-down movements in the lattice. However, if at each iteration we are able to select the best candidate bit and to avoid conflicts, we only allow steps down in the Boolean lattice. We use the heuristic described in Section 3.4.2 to choose the best candidate bit to flip-off.

A practical example. Consider again the example in Table A.1. Since at the beginning $\mathcal{S} = \mathcal{D}_+$, we will only report $|\mathcal{D}_{+i}^{\,0}|$. For the first positive record $\chi_1 = 110\ 01$, we have: $\mathcal{D}_{-1}^{\,0} = \{01101, 01110\}$, $\mathcal{D}_{-2}^{\,0} = \{10101\}$, $\mathcal{D}_{-5}^{\,0} = \{01110\}$. We have therefore: $d_l(\chi_1, \mathcal{D}_{-1}^{\,0}) = 1$, $d_l(\chi_1, \mathcal{D}_{-2}^{\,0}) = 1$, $d_l(\chi_1, \mathcal{D}_{-5}^{\,0}) = 2$. We already know that flipping-off either the first or the second bit to 0 would lead to a conflict: thus, we directly flip-off the fifth bit to obtain the boundary point 110 00, independently from the value of $|\mathcal{D}_{+5}^{\,0}|$. Element 110 00 is added in the set of boundary points $\mathcal{A}$.

For the second positive record $\chi_3 = 101\ 10$, we have: $\mathcal{D}_{-1}^{\,0} = \{01101, 01110\}$, $\mathcal{D}_{-3}^{\,0} = \emptyset$, $\mathcal{D}_{-5}^{\,0} = \{01110\}$. We have therefore: $d_l(\chi_1, \mathcal{D}_{-1}^{\,0}) = 1$, $d_l(\chi_1, \mathcal{D}_{-3}^{\,0}) = undefined$, and $d_l(\chi_1, \mathcal{D}_{-4}^{\,0}) = 1$. Although $i = 3$ induces a distance from an empty set, since we know that flipping-off other indexes generates conflicts, we can immediately label 100 10 as boundary point and add it to $\mathcal{A}$.

From boundary set to rules. At the end of the previous phase, we obtain the boundary set $\mathcal{A} = \{11000, 10010\}$. In this case, each boundary point covers only one distinct positive sample, therefore the union of the two points covers all the set of positive samples and both points are kept after the regularization. Suppose to follow a positive set cover strategy, without early stopping condition. Then, the boundary set can be immediately mapped to the rule set shown in Figure A.2.

IF CPU $\in [95, max]$
OR CPU $\in [81, max]$ AND MEM $\in [85, max]$
THEN Label $= 1$
ELSE Label $= 0$

Figure A.2: Rule set extracted from the boundary $\mathcal{A}$.

Differences with Muselli and Quarati (2005). Our base method is similar to Muselli and Quarati (2005) that we took as a reference since it presents interesting properties: it is a bottom-up method, easily parallelizable, incorporating interpretability goals in the induction process. To better analyze the differences with our proposal, we refer to a technical report (Muselli and Quarati, 2004) where the authors provide more details about their *shadow clustering* algorithm. Looking at the pseudo-code (Figure 4 of Muselli and Quarati (2004)) we can firstly notice that our heuristics H_1 and H_2 are 3-length tuples (instead of 4-length) where we avoid to compute what Muselli and Quarati (2004) indicate as $|B_i^1|$ (we verified experimentally that it has no impact in practice). The main improvement, however, is related to the computation of the distance $d_l(t(I \cup J), \mathcal{D}_{-i}^{\,0})$ which complexity is $O(nd)$. In Muselli and Quarati (2004), such distance is computed within a while loop (of length d in the worst case), for every candidate

bit $i \in I$ to flip. We instead compute the distance only once outside the while loop, and update it iteratively for i_{best} only (the complexity of the update is $O(n)$ instead of $O(nd)$). Indeed, we flip one bit at a time and there is no need to re-compute the distance from scratch for the other bits. Thus, we lower the total complexity from $O(n^2d^3)$ to $O(n^2d^2)$.

A.2 Parallel and Distributed Implementation

LIBRE is amenable to parallel and distributed implementations. Indeed, it processes one positive sample at a time. An exhaustive version of the FINDBOUNDARYPOINT procedure is embarrassingly parallel and it is easily parallelizable on multi-core architectures: it is sufficient to spawn a UNIX process per positive sample, and exploit all available cores.

In contrast, the approximate procedure requires a slightly more involved approach. Indeed, the approximate FINDBOUNDARYPOINT procedure processes positive records that have not yet been covered by any boundary point. Hence, a global view on the set $\mathcal{S}$ is required. We experimented with two alternatives. The first is to place $\mathcal{S}$ in a shared, in-RAM datastore, as UNIX processes – unlike threads – do not have shared memory access. The second alternative is to simply let each individual process to hold their own version of $\mathcal{S}$, thus sacrificing a global view. Our experiments indicate that the loss in performance due to a local view only is negligible, and largely out-weighted by the gain in performance, since the execution time decreases linearly with the number of spawned UNIX processes. Moreover, both $\mathcal{D}_+$ and $\mathcal{D}_-$ remain consistent throughout the whole induction phase.

LIBRE can be easily distributed such that it can run on a cluster of machines, using for example a distributed computing framework such as Apache Spark (Zaharia et al., 2016). This approach, called *data parallelism*, splits input data across machines, and let each machine execute, independently, a weak learner. The data splitting operation shuffles random subsets of the input features to each worker machine. Once each worker finishes to generate the local rule sets, they are merged in the "driver" machine, which eventually applies the filtering and executes the rule selection procedure to produce the final boundary.

A.3 Scalability Evaluation

Here, we extensively test the scalability of LIBRE. We use up to 50 features and investigate also the impact of class imbalance on the execution time.

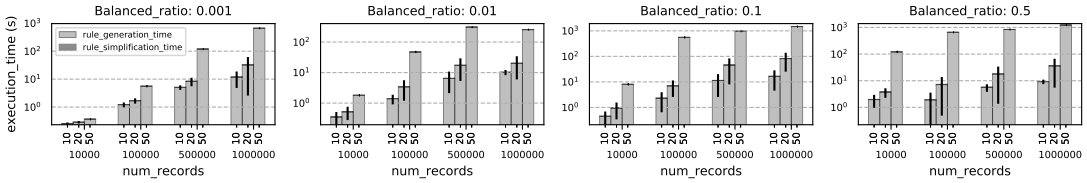


Figure A.3: Run time on synthetic data.

Synthetic Dataset. For the scalability evaluation, we synthetically generate a dataset with $1'000'000$ records and 50 continuous features with randomly generated values in the domain $[0, 100]$. Then, we randomly generate four sets of binary labels with a class imbalance ratio of 0.001, 0.01, 0.1, and 0.5 respectively.

Settings. We vary the number of records (10'000, 100'000, 500'000, 1'000'000), features (10, 20, 50), and class imbalance ratio (0.001, 0.01, 0.1, 0.5): for each dataset configuration, LIBRE runs up to 100 times with different randomly generated subsets of features of size 10, 20, and 50; the average execution time in seconds is reported as a sum of two contributions: rule generation and simplification times. Times refer to one weak learner only: if N weak learners run in parallel, the reported time is still a good estimate. Before executing LIBRE, we discretize the dataset with a discretization threshold equal to 6, that we empirically find out to be a good value. The simplification procedure runs on the top 500 rules, if more are generated.

Results. As shown in Figure A.3, the execution time is dominated by the rule generation term. Given a class imbalance ratio, execution time increases as long as we increase the number of records and features. The generation time also depends on which features are fed into the model for two main reasons: i) ChiMerge encodes bad predictive features with bigger domains, increasing the search space; ii) the generation procedure will struggle more to generate rules when it runs on features that are not that useful to predict the target class. This explains the high variance in the results. Intuitively, as long as the class imbalance ratio gets close to 0.5, the number of processed records increases, together with the execution time. However, we experimentally verified that this effect is somehow compensated by the higher number of negative records. We run the rule generation procedure up to 50 features just for experimental purposes: for practical applications, if interpretability is a need, it is more convenient to limit the number of features and train an ensemble with more learners in order to generate compact rules in a reasonable time.

A.4 Interpretable rule sets

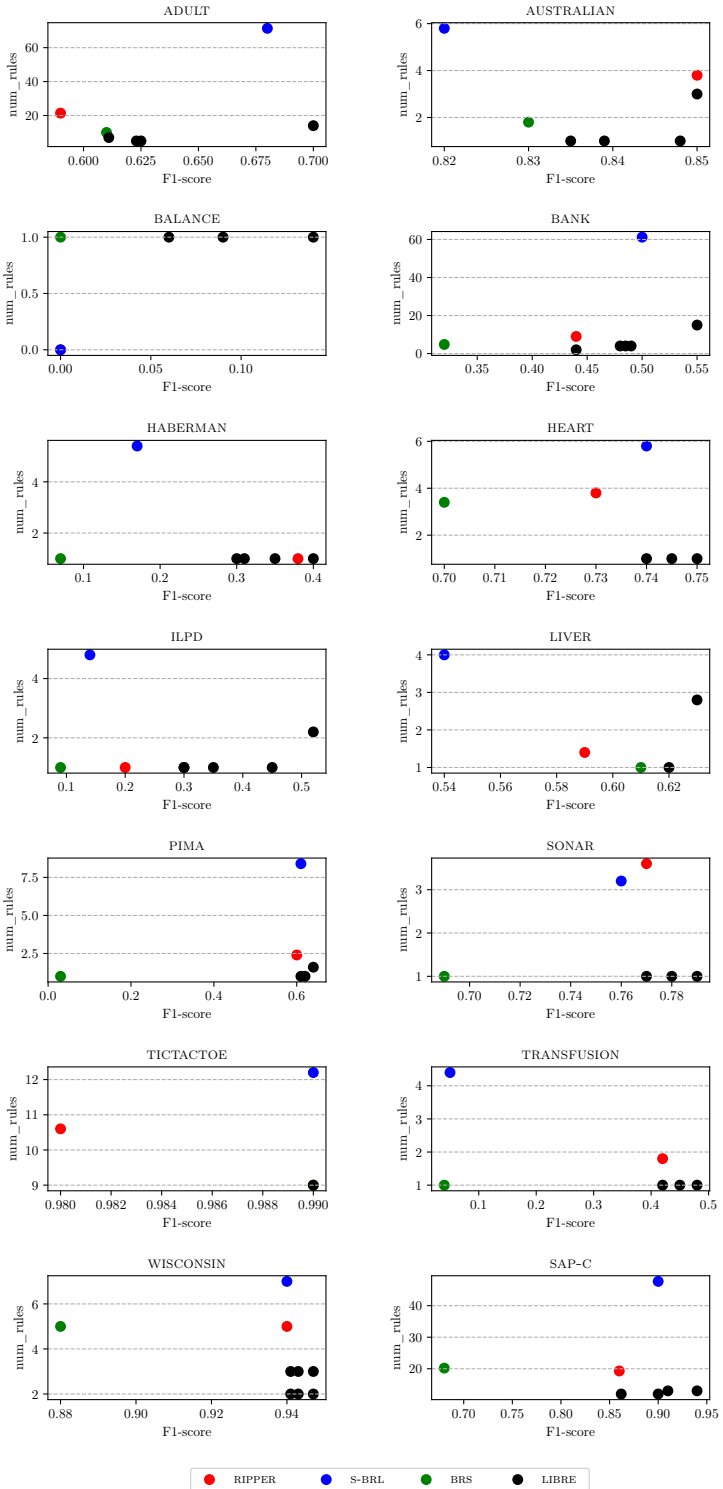


Figure A.4: F1-score vs number of rules.

A.5 More Examples of Rule Sets learned by LIBRE

In this section, we report additional examples for the medical UCI datasets described in Table 3.1[1], for which it might be interesting to understand the relation between input features and the predicted diseases.

IF number_of_positive_axillary_nodes $\in [2, max]$
THEN died within 5 years
ELSE survived 5 years or longer

Figure A.5: Example of rule set learned by LIBRE for HABERMAN.

IF (slope_of_the_peak_exercise $\in \{flat, downsloping\}$ AND number_of_major_vessels $\in [1, 3]$)
OR (chest_pain_type $\in \{asymptomatic\}$ AND thal $\in \{reversable\_defect\}$)
OR (sex $\in \{male\}$ AND fasting_blood_sugar_>120mg/dl $\in \{False\}$ AND
number_of_major_vessels $\in [1, 3]$)
THEN class = presence
ELSE class = absence

Figure A.6: Example of rule set learned by LIBRE for HEART.

IF (TB $\in [min, 2)$ AND sgbp $\in [min, 42)$)
OR (TB $\in [min, 2)$ AND alkphos $\in [min, 184)$)
OR (age $\in [35, 39), [56, 57)$ AND sgbp $\in [42, 148)$)
THEN class = liver patient
ELSE class = no liver partient

Figure A.7: Example of rule set learned by LIBRE for ILPD.

IF mean_corpuscular_volume $\in [90, 96)$
OR gamma_glutamyl_transpeptidase $\in [20, max]$
THEN liver_disorder = True
ELSE liver_disorder = False

Figure A.8: Example of rule set learned by LIBRE for LIVER.

[1]Different rule sets may be obtained depending on how folds are randomly built during cross validation.

IF (glucose $\in [158, max]$ AND blood_pressure $\in [56, max]$)
OR (glucose $\in [110, 158]$ AND BMI $\in [30.7, max]$)
OR (pregnancies $\in [4, max]$ AND diabetes_predigree_func $\in [0.529, max]$)
THEN diabetes = True
ELSE diabetes = False

Figure A.9: Example of rule set learned by LIBRE for PIMA.

IF (months_since_last_donation $\in [0, 8)$ AND total_blood_donated $\in [1250, max]$)
THEN transfusion = Yes
ELSE transfusion = No

Figure A.10: Example of rule set learned by LIBRE for TRANSFUSION.

IF uniformity_of_cell_shape $\in [5, max]$
OR (clump_thickness $\in [2, max]$ AND bare_nuclei $\in [8, max]$)
OR (clump_thickness $\in [7, max]$ AND marginal_adhesion $\in [1, 2), [4, max]$)
THEN malignant = True
ELSE malignant = False

Figure A.11: Example of rule set learned by LIBRE for WISCONSIN.

APPENDIX

B

ADDITIONAL DETAILS AND RESULTS ABOUT IDVAE

B.1 ELBO derivation for IDVAE

$$\begin{aligned}
\log p(\mathbf{x}, \mathbf{u}) &= \log \int p(\mathbf{x}, \mathbf{u}, \mathbf{z}) d\mathbf{z} = \\
&= \log \int p(\mathbf{x}|\mathbf{u}, \mathbf{z}) p(\mathbf{z}|\mathbf{u}) p(\mathbf{u}) d\mathbf{z} = \\
&= \log \int \frac{p(\mathbf{x}|\mathbf{u}, \mathbf{z}) p(\mathbf{z}|\mathbf{u}) p(\mathbf{u})}{q(\mathbf{z}|\mathbf{x}, \mathbf{u})} q(\mathbf{z}|\mathbf{x}, \mathbf{u}) d\mathbf{z} \geq \\
&\geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\mathbf{u})}[\log \frac{p(\mathbf{x}|\mathbf{u}, \mathbf{z}) p(\mathbf{z}|\mathbf{u}) p(\mathbf{u})}{q(\mathbf{z}|\mathbf{x}, \mathbf{u})}] = \\
&= \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\mathbf{u})}[\log p(\mathbf{x}|\mathbf{u}, \mathbf{z})] - KL(q(\mathbf{z}|\mathbf{x}, \mathbf{u})||p(\mathbf{z}|\mathbf{u})) + \log p(\mathbf{u}), \qquad \text{(B.1)}
\end{aligned}$$

where:

$$\begin{aligned}
\log p(\mathbf{u}) &= \log \int p(\mathbf{u}, \mathbf{z}) d\mathbf{z} \geq \\
&\geq \mathbb{E}_{q(\mathbf{z}|\mathbf{u})}[\log p(\mathbf{u}|\mathbf{z})] - KL(q(\mathbf{z}|\mathbf{u})||p(\mathbf{z})). \qquad \text{(B.2)}
\end{aligned}$$

B.2 ELBO derivation for SS-IDVAE

$$\begin{aligned}
\log p(\mathbf{x}) &= \log \int p(\mathbf{x}, \mathbf{u}, \mathbf{z}) d\mathbf{u} d\mathbf{z} = \\
&= \log \int p(\mathbf{x}|\mathbf{u}, \mathbf{z}) p(\mathbf{z}|\mathbf{u}) p(\mathbf{u}) d\mathbf{u} d\mathbf{z} =
\end{aligned}$$

$$
\begin{aligned}
&= \log \int \frac{p(\mathbf{x}|\mathbf{u},\mathbf{z})p(\mathbf{z}|\mathbf{u})p(\mathbf{u})}{q(\mathbf{u},\mathbf{z}|\mathbf{x})} q(\mathbf{u},\mathbf{z}|\mathbf{x}) d\mathbf{u} d\mathbf{z} \geq \\
&\geq \mathbb{E}_{q(\mathbf{u},\mathbf{z}|\mathbf{x})}[\log \frac{p(\mathbf{x}|\mathbf{u},\mathbf{z})p(\mathbf{z}|\mathbf{u})p(\mathbf{u})}{q(\mathbf{u},\mathbf{z}|\mathbf{x})}] = \\
&= \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\mathbf{u})q(\mathbf{u}|\mathbf{x})}[\log \frac{p(\mathbf{x}|\mathbf{u},\mathbf{z})p(\mathbf{z}|\mathbf{u})p(\mathbf{u})}{q(\mathbf{z}|\mathbf{x},\mathbf{u})q(\mathbf{u}|\mathbf{x})}] = \\
&= \mathbb{E}_{q(\mathbf{u}|\mathbf{x})}[\mathcal{L}_{\text{IDVAE}}] + \mathcal{H}(q(\mathbf{u}|\mathbf{x})). \qquad \text{(B.3)}
\end{aligned}
$$

By combining Equations B.1 to B.3 we obtain $\mathcal{L}_{\text{SS-IDVAE}}$, where it is clear that we use the sum over the data samples instead of the expectation. As stated in the Section 5.4.3, we add a term – $\mathbb{E}_{(\mathbf{x},\mathbf{u})\sim p_l}[\log q(\mathbf{u}|\mathbf{x})]$ – such that it can learn also from labeled data.

B.3 Proof of Theorem 1

In this section, we report the proof of Theorem 1, following the same strategy of Khemakhem et al. (2020). The proof consists of three main steps.

In the first step, we use assumption (i) to demonstrate that observed data distributions are equal to noiseless distributions. Supposing to have two sets of parameters $(\mathbf{f}, \mathbf{T}, \boldsymbol{\eta})$ and $(\tilde{\mathbf{f}}, \tilde{\mathbf{T}}, \tilde{\boldsymbol{\eta}})$, with a change of variable $\bar{\mathbf{x}} = \mathbf{f}(\mathbf{z}) = \tilde{\mathbf{f}}(\mathbf{z})$, we show that:

$$
\tilde{p}_{\mathbf{T},\boldsymbol{\eta},\mathbf{f},\mathbf{u}}(\mathbf{x}) = \tilde{p}_{\tilde{\mathbf{T}},\tilde{\boldsymbol{\eta}},\tilde{\mathbf{f}},\tilde{\mathbf{u}}}(\mathbf{x}), \qquad \text{(B.4)}
$$

where:

$$
\tilde{p}_{\mathbf{T},\boldsymbol{\eta},\mathbf{f},\mathbf{u}}(\mathbf{x}) = p_{\mathbf{T},\boldsymbol{\eta}}(\mathbf{f}^{-1}(\mathbf{x})|\mathbf{u})|\det J_{\mathbf{f}^{-1}}(\mathbf{x})|\mathbb{1}_{\mathcal{X}}(\mathbf{x}) \qquad \text{(B.5)}
$$

In the second step, we use assumption (iv) to remove all the terms that are a function of $\mathbf{x}$ or $\mathbf{u}$. By substituting $p_{\mathbf{T},\boldsymbol{\eta}}$ with its exponential conditionally factorial form, taking the log of both sides of Equation B.5, we obtain $dk + 1$ equations. Then:

$$
\mathbf{T}(\mathbf{f}^{-1}(\mathbf{x})) = \mathbf{A}\mathbf{T}'(\mathbf{f}'^{-1}(\mathbf{x})) + \mathbf{c}. \qquad \text{(B.6)}
$$

In the last step, assumptions (i) and (iii) are used to show that the linear transformation is invertible and so $(\mathbf{f}, \mathbf{T}, \boldsymbol{\eta}) \sim (\tilde{\mathbf{f}}, \tilde{\mathbf{T}}, \tilde{\boldsymbol{\eta}})$. This concludes the proof.

For further details, we point the reader to section **B** of the supplement in Khemakhem et al. (2020), which holds also for our variant of the theorem.

B.4 Model architectures and parameters

All the selected methods (including the semi-supervised variants) share the same convolutional architecture. The conditional prior in IVAE is a MLP network, in IDVAE we use a simple MLP VAE, both with leaky ReLU activation functions. The ground-truth factor learner implementing $q_\zeta(\mathbf{u}|\mathbf{x})$ in SS-IDVAE and SS-IVAE is a CNN.

Encoder	**Decoder**
Input: $64 \times 64 \times$ number of channels	Input: $\mathbb{R}^d$, where d is the number of ground-truth factors
4×4 conv, 32 ReLU, stride 2	FC, 256 ReLU
4×4 conv, 32 ReLU, stride 2	FC, $4 \times 4 \times 64$ ReLU
4×4 conv, 64 ReLU, stride 2	4×4 upconv, 64 ReLU, stride 2
4×4 conv, 64 ReLU, stride 2	4×4 upconv, 32 ReLU, stride 2
FC 256*, FC $2 \times d$	4×4 upconv, 32 ReLU, stride 2
	4×4 upconv, number of channels, stride 2

Table B.1: Main Encoder-Decoder architecture. In IVAE and IDVAE, we give $\mathbf{u}$ as input to the fully connected layer of the Encoder which size becomes $256 + d$.

Conditional Prior Encoder	**Conditional Prior Decoder**
FC, 1000 leaky ReLU	FC, 1000 leaky ReLU
FC, 1000 leaky ReLU	FC, 1000 leaky ReLU
FC, 1000 leaky ReLU	FC, 1000 leaky ReLU
FC $2 \times d$	FC d

Table B.2: IDVAE Conditional Prior Encoder-Decoder architecture. IVAE uses the encoder only.

Ground-truth Factor Learner
Input: $64 \times 64 \times$ number of channels. d is the number of ground-truth factors.
4×4 conv, 32 ReLU, stride 2
4×4 conv, 32 ReLU, stride 2
4×4 conv, 64 ReLU, stride 2
4×4 conv, 64 ReLU, stride 2
FC 256, FC $2 \times d$

Table B.3: Ground-truth factor learner implementing $q_\zeta(\mathbf{u}|\mathbf{x})$ in SS-IDVAE and SS-IVAE.

Parameters	**Values**
batch_size	64
optimizer	Adam
Adam: beta1	0.9
Adam: beta2	0.999
Adam: epsilon	1e-8
Adam: learning_rate	1e-4
training_steps	300'000

Table B.4: Common hyperparameters to each of the considered methods.

B.5 Implementation of disentanglement metrics

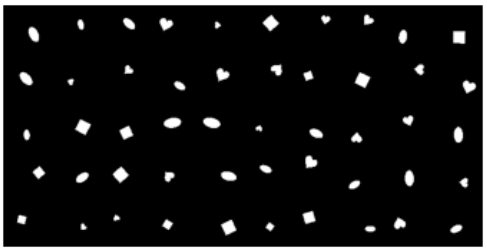
As explained in Section 5.5.1, the implementation of the selected disentanglement evaluation metrics is based on Locatello et al. (2019b). We report the main parameters in Table B.5.

Disentanglement metrics	Parameters
Beta score	train_size=10'000, test_size=5'000, batch_size=64, predictor=logistic_regression
MIG	train_size=10'000, n_bins=20
Modularity and Explicitness	train_size=10'000, test_size=5'000, batch_size=16, n_bins=20
SAP score	train_size=10'000, test_size=5'000, batch_size=16, predictor=linearSVM, C=0.01

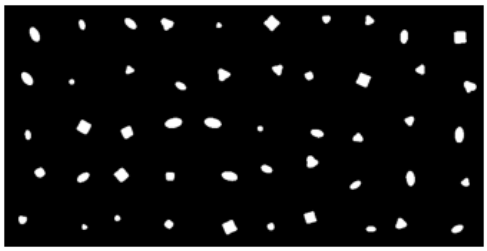
Table B.5: Disentanglement metrics and their parameters.

B.6 Full experiments

In this section, we illustrate the full set of experiments, including reconstructions and latent traversals.



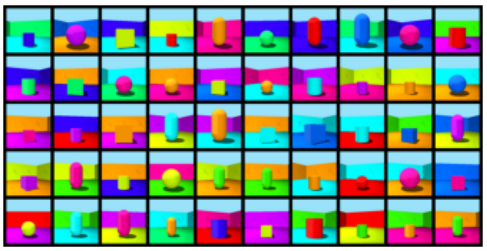
(a) DSPRITES: original observations.



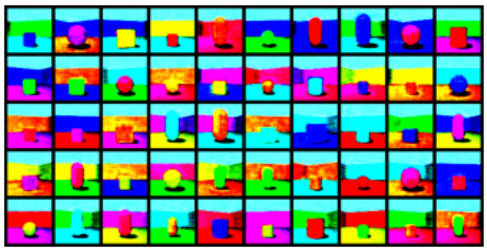
(b) DSPRITES: reconstructions by IDVAE.

(c) CARS3D: original observations.

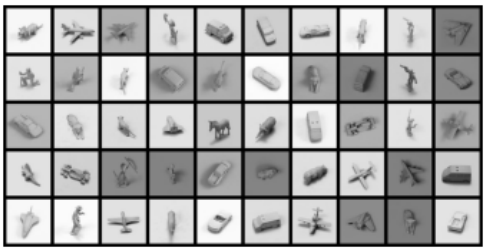
(d) CARS3D: reconstructions by IDVAE.



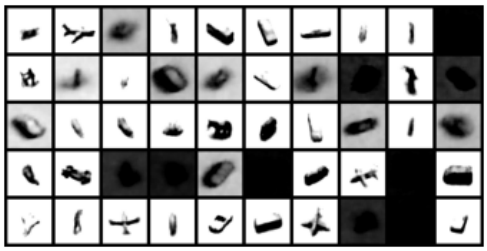
(e) SHAPES3D: original observations.



(f) SHAPES3D: reconstructions by IDVAE.

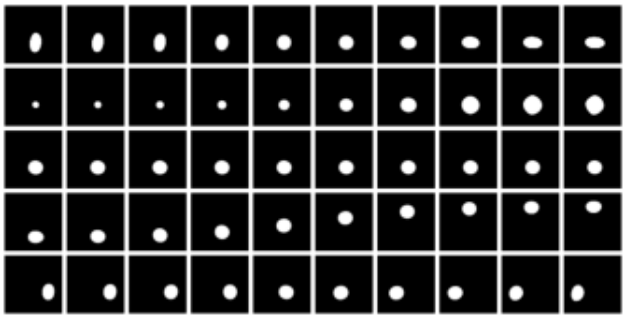


(g) SMALLNORB: original observations.

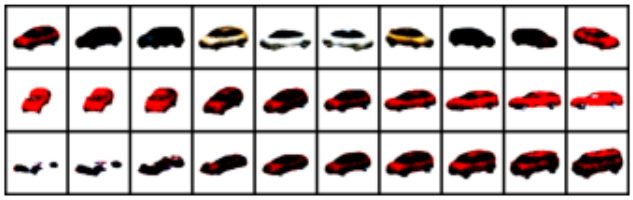


(h) SMALLNORB: reconstructions by IDVAE.

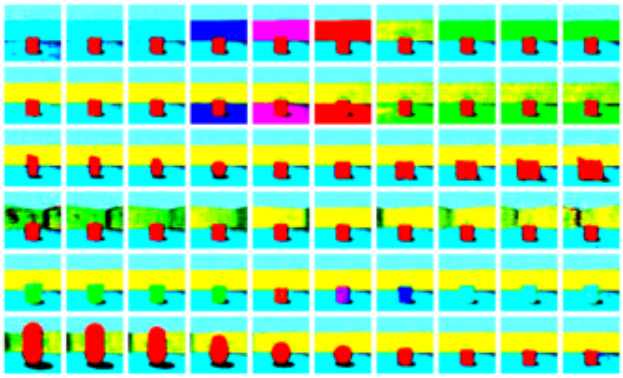
Figure B.1: Original observations vs IDVAE reconstructions.



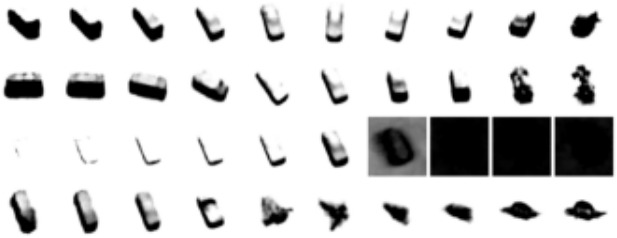
(a) DSPRITES.



(b) CARS3D.



(c) SHAPES3D.



(d) SMALLNORB.

Figure B.2: IDVAE latent traversals. Each row corresponds to a dimension of $\mathbf{z}$, that we vary in the range $[-3, 3]$. We can see that, in some cases, changing a dimension can affect multiple ground-truth factors, meaning that IDVAE has not obtained full disentanglement. (a) From top to bottom: orientation, scale, shape(?), posY, posX. (b) From top to bottom: azimuth, elevation, object type. (c) From top to bottom: wall color, floor color, object type, azimuth, object color, object size. (d) azimuth, elevation, lighting, category.

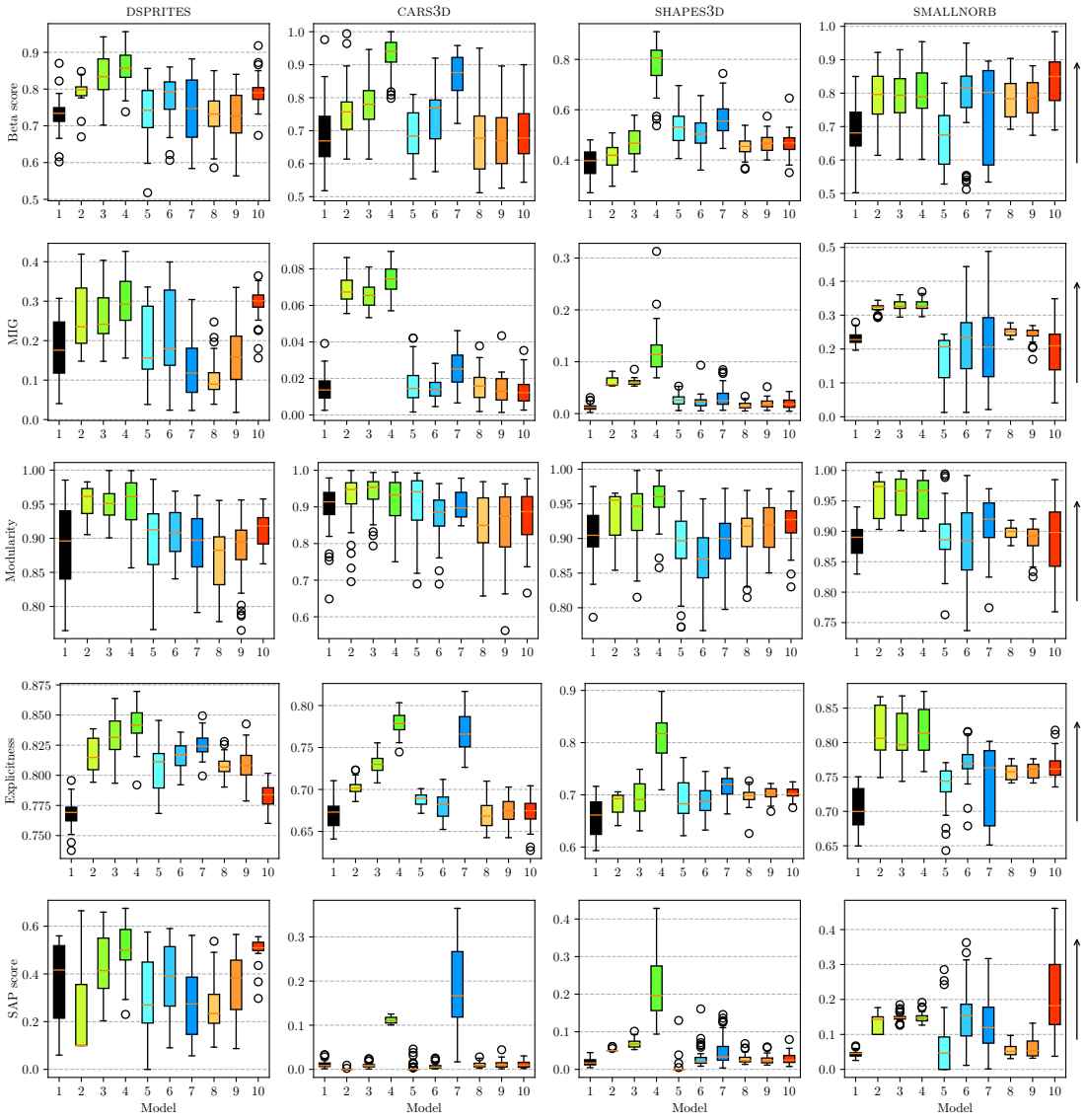


Figure B.3: Beta score, MIG, Modularity, Explicitness, and SAP (the higher the better). 1=β-VAE, 2=SS-IDVAE (1%), 3=SS-IDVAE (10%), 4=IDVAE, 5=SS-IVAE (1%), 6=SS-IVAE (10%), 7=IVAE, 8=SS-FULLVAE (1%), 9=SS-FULLVAE (10%), 10=FULLVAE. Percentage of labeled samples in parenthesis.

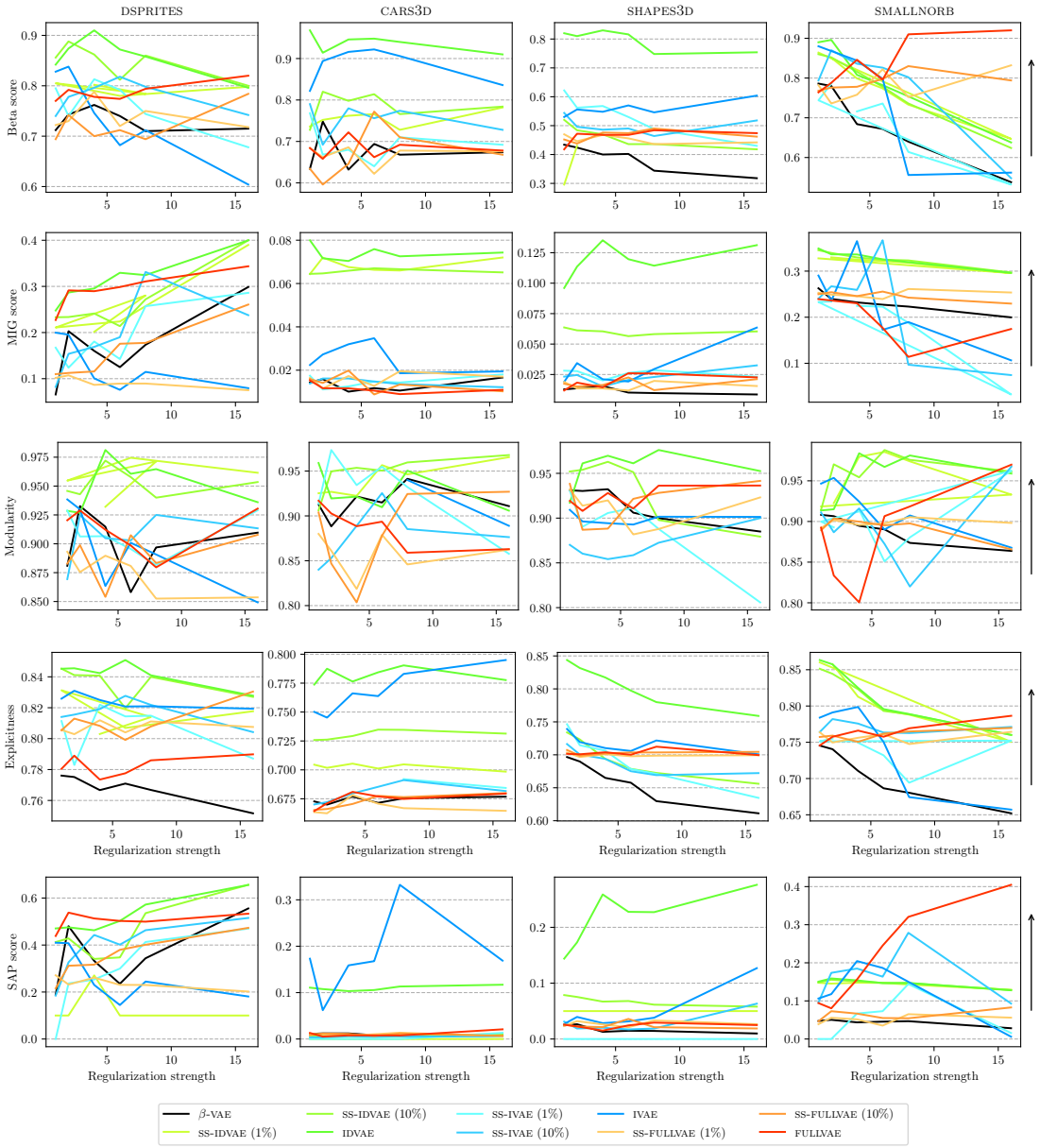


Figure B.4: Beta score, MIG, modularity, explicitness and SAP median (the higher the better) as a function of the regularization strength, for each method on DSPRITES, CARS3D, SHAPES3D, SMALLNORB.