UNIVERSITY OF
CAMBRIDGE

Deep concept reasoning: beyond the accuracy-interpretability trade-off

Pietro Barbiero

Clare College

This dissertation is submitted for the degree of Doctor of Philosophy

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Preface and specified in the text. It is not substantially the same as any that I have submitted, or am concurrently submitting, for a degree or diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. I further state that no substantial part of my dissertation has already been submitted, or is being concurrently submitted, for any such degree, diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. This dissertation does not exceed the prescribed limit of 60 000 words.

Pietro Barbiero
May, 2023

Abstract

Deep concept reasoning: beyond the accuracy-interpretability trade-off

Pietro Barbiero

Deep learning researchers stockpile ground-breaking achievements almost as fast as they find flaws in their models. Although deep learning models can achieve superhuman performances, explaining deep learning decisions and mistakes is often impossible even for "explainable AI" specialists, causing lawmakers to question the ethical and legal ramifications of deploying deep learning systems. For this reason, the key open problem in the field is to increase deep neural networks transparency and trustworthiness to enable a safe deployment of such technologies.

The lack of human trust in deep learning is affected by three key factors. Firstly, the absence of a formal and comprehensive theory undermines the field of explainable AI. This leads to ill-posed questions, induces re-discovery of similar ideas, and impedes researchers to approach the domain. Secondly, the explainable AI literature is mostly dominated by methods providing post-hoc, qualitative, and local explanations, which are often inaccurate and misleading. Finally, machine learning systems—including deep neural networks—struggle in striking a balance between task accuracy and interpretability. Existing solutions either sacrifice model transparency for task accuracy or vice versa, making it difficult to optimize both objectives simultaneously.

This thesis includes four research works contributing in addressing these challenges. The first work addresses the lack of a formal theory of explainable AI. This work proposes the first-ever theory of explainable AI and concept learning, which formalizes some of the fundamental ideas used in this field. The key innovation of this chapter is the use of categorical structures to formalize explainable AI notions and processes. The use of category theory is particularly noteworthy as it provides a sound and abstract formalism to examine general structures and systems of structures, avoiding contingent details and focusing on their fundamental essence. This theoretical foundation serves as a solid basis for the other chapters in the thesis. The second work aims to overcome the limitations of current explainable AI techniques providing post-hoc, qualitative, and local explanations.

To this end, this work proposes Logic Explained Networks, a novel class of concept-based models that can solve and explain classification problems simultaneously. The key innovation of Logic Explained Networks is a sparse attention layer that selects the most relevant concepts in neural concept-based models. This way, the model learns to generate simple logic explanations. The third work tackles the accuracy-explainability trade-off, a major limitation of concept-based models. To address this issue, this work proposes Concept Embedding Models. The key innovation of Concept Embeddings Models is a fully supervised high-dimensional concept representation. The high-dimensional representation enables Concept Embedding Models to overcome the information bottleneck, enabling them to achieve state-of-the-art accuracy without sacrificing model transparency. The fourth work addresses the limitations of Concept Embeddings Models which are unable to provide concept-based logic explanations for their predictions. To fill this gap, this work presents the Deep Concept Reasoner, the first interpretable concept-based model using concept embeddings. The key innovation of the Deep Concept Reasoner is the use of neural networks to generate interpretable rules which are executed symbolically to make task predictions. This enables the Deep Concept Reasoner to attain state-of-the-art performance in complex tasks and to provide human-understandable and formal explanations for its predictions.

Overall, this thesis makes significant contributions by introducing the first formal theory of explainable AI and presenting novel deep learning techniques going beyond the current accuracy-interpretability trade-off. The results of the experiments demonstrate how these innovations lead to a new generation of deep learning architectures that are both transparent and accurate. The introduction of these new techniques lays the groundwork to increase deep learning transparency and trustworthiness, enabling a safe deployment of robust and controllable machine learning agents.

To my beloved wife.
Pietro

Acknowledgements

To my beloved wife, Betta, whose unwavering love has been my rock throughout this incredible journey, I am forever grateful. Thank you for bringing immense joy, entertainment, and ease into my life through your constant presence. I humbly ask for your forgiveness for the countless times I interrupted your sleep with new ideas in the middle of the night, for the late nights preceding conference deadlines, and for my limited contributions at home during the busiest periods. I acknowledge that without your support, I could not have accomplished even half of what I have achieved. You are my pillar of strength.

To Pietro (Lió, yes, not myself), the most extraordinary supervisor anyone could hope for. Without you, this work, this incredible experience, would have never even begun. You believed in me, you took a chance on me, and bestowed upon me this once-in-a-lifetime opportunity. You have opened up a world of possibilities for me, and I can never adequately express my appreciation. So let me begin by saying, 'thank you!'.

To all my friends who shared my passion for AI and accompanied me on this remarkable journey, I express my heartfelt thanks. Adrian, Alberto, Alex, Alisia, Andrea, Andrei, Antonio, Botty, Chaitanya, Charlotte, Davide, Dmitry, Dobrik, Donato, Elena, Emanuele, Emma, Fabrizio, Federico, Francesco, Francesco, Frédéric, Gabriele, Gabriele, Gianluca, Giansalvo, Giovanni, Giulio, Giuseppe, Han, Iulia, Jacob, Lorenzo, Marco, Maria Sofia, Mateo, Mateja, Michelangelo, Nicolò, Nikola, Oghuzan, Paul, Petar, Ramon, Rishabh, Simon, Stefano, Steve, Thiago, Vincenzo, Zohreh—your names are etched in my heart. A special thank you to Francesco e Gabriele, who, more than anyone else, bore the brunt of my darkest moments of annoyance and madness. Thank you for standing by my side, despite my continuous mistakes. This work belongs to you as much as it belongs to me.

Alla mia famiglia, ai miei genitori, Giuseppe e Marta, ai miei fratelli, Silvia e Taddeo, ai miei nonni, Adriana, Cesare, Claudio, Giulio e Grazia, ai miei cugini, ai miei zii, "Indiani", Piemontesi, Romani, Toscani, Veneti, Umbri, grazie per il vostro amore, il vostro sostegno, le vostre parole di conforto e di incoraggiamento. Grazie per esservi presi cura di me. Grazie per il cibo, i giochi e le discussioni vivaci. Grazie per tutto ciò che mi avete insegnato. La persona che sono è frutto del vostro duro lavoro. Questa tesi e questo lavoro non avrebbero neanche avuto inizio senza di voi.

Ai miei amici di sempre Edo, Gabri, Vitto, a voi a cui posso dire tutto, a voi che mi

fate tornare ai fondamentali concreti della vita di cui questo lavoro rappresenta solo una visione passeggera.

Contents

Chapter 1

Introduction

1.1 Lack of human trust: the key open problem in deep learning

Deep learning researchers stockpile ground-breaking achievements almost as fast as they find (consistently similar!) flaws in their models (Marcus et al., 2022). On the one hand, the extremely high learning capacity may allow deep learning to achieve super-human performances on some tasks. Thanks to this ability, deep learning is already spreading and generating a strong impact in fields like medicine, chemistry, physics, and social networks as companies started integrating deep learning tools in products for cancer detection, drug development, DNA analysis, particles' trajectory prediction, or fake-news detection. On the other hand, this high learning capacity comes at the cost of making impossible even for researchers to trace back and explain incorrect predictions. This represents a significant limitation in real-world applications as it does not allow human experts to interpret and use deep learning mistakes to improve the model and deploy better solutions.

As this trend got worse, lawmakers started questioning the ethical (Durán and Jongsma, 2021; Lo Piano, 2020) and legal (Wachter et al., 2017; EUGDPR, 2017) ramifications of the deployment of deep learning systems. Philosophical concerns turn into pressing needs in safety-critical domains which require accurate and trustworthy AI agents (Rudin, 2019; Shen, 2022). As a response, the research community intensified the effort in developing trustworthy, fair and reliable models. This effort led to relevant innovations aiming at explaining the inner workings of deep neural networks. However, after years of research, trustworthy deep learning models have still several flaws and are yet to be realized.

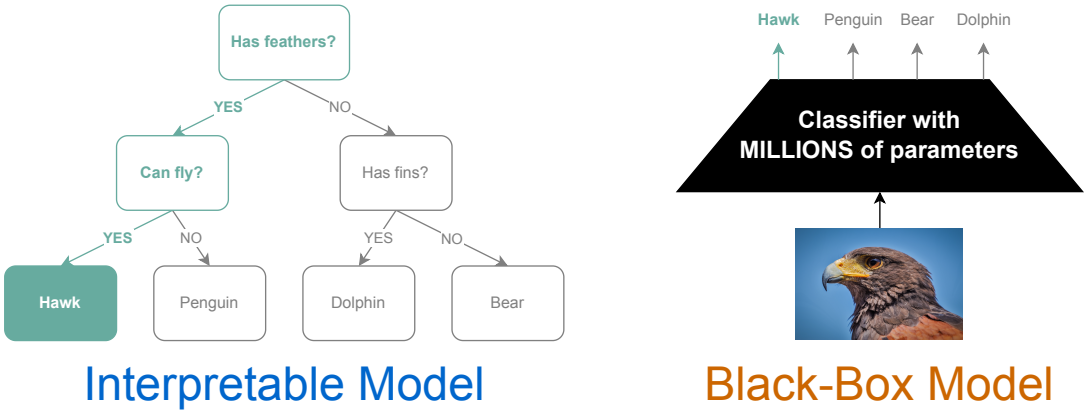


Figure 1.1: Visual examples of interpretable (left) and black-box (right) models. Interpretable models (such as decision trees) are inherently transparent and humans can easily understand their decision process. On the contrary, the use of millions of parameters in black-box models (such as deep convolutional networks) hinders the exact and straightforward understanding of their decision process.

1.2 Knowledge gaps interfering with human trust

There are several factors which currently limit human trust in deep learning systems. Here I describe the main knowledge gaps which I focus on in this work.

1.2.1 Deep learning systems are not interpretable

One of the main reasons why humans question deep learning is a lack of understanding of the decision process of these machines. To address this, researchers in deep learning started developing *interpretabile* and *explainable* methods (Figure 1.1).

Interpretable methods. Even if there are no common accepted formal definitions, a model is considered interpretable when its decision process is generally transparent and can be understood directly by its structure and parameters (see Figure 1.1, left). Many classical machine learning techniques are devised to be intrinsically interpretable. Prominent examples are: Logistic Regression (McKelvey and Zavoina, 1975), Generalized Additive Models (Hastie and Tibshirani, 1987; Lou et al., 2012; Caruana et al., 2015), Decision Trees (Breiman et al., 1984; Quinlan, 1986, 2014), Decision Lists (Rivest, 1987; Letham et al., 2015; Angelino et al., 2018), and Bayesian Rule Lists (Letham et al., 2015). However, most of them struggle in solving complex classification problems. Logistic regression or a perceptron, for instance, in their vanilla definition, can only recognize linear patterns and they cannot solve even simple non-linear problems such as the exclusive OR of two inputs (Minsky and Papert, 1969).

Explainable methods. On the other side of the spectrum, "black-box" models (see Figure 1.1, right) represent some of the most powerful machine learning systems which are able to solve incredibly complex tasks. However, the complexity of these models makes the exact understanding of their decision process utterly impossible for humans. For this reason, researchers started developing explanation methods to provide simple "surrogate" explanations for trained "black-box" models. Most methods focus on identifying and ranking the most relevant input features (Erhan et al., 2010; Simonyan et al., 2013; Zeiler and Fergus, 2014; Ribeiro et al., 2016b,a; Lundberg and Lee, 2017; Selvaraju et al., 2017). Feature scores are usually computed sample by sample (i.e. providing *local explanations*) analyzing the activation patterns in the hidden layers of neural networks (Erhan et al., 2010; Simonyan et al., 2013; Zeiler and Fergus, 2014; Selvaraju et al., 2017) or by following a model-agnostic approach (Ribeiro et al., 2016a; Lundberg and Lee, 2017). To enhance human understanding of feature scoring methods, concept-based approaches have been effectively employed for identifying common activations patterns in the last nodes of neural networks corresponding to human categories (Kim et al., 2018; Kazhdan et al., 2020) or constraining the network to learn such concepts (Chen et al., 2020; Koh et al., 2020). However, the main issue of explaining "black-box" is that the explanations extracted with a surrogate model may not be perfectly faithful to the original model as extensively discussed by Rudin (2019). Indeed, the extraction of explanations often requires a form of model simplification which may significantly mislead human users. This misalignment between explanations and actual model behavior is one of the main reason why explainable models have been harshly criticized, especially when they are used for high-stakes decisions (Rudin, 2019). In these contexts, interpretable models are way more robust and trustworthy as their behavior does not require further explanations, thus preventing all forms of misalignment. Unfortunately, current deep learning systems are miles away from being interpretable, and, consequently, trustworthy (Rudin, 2019).

Example 1.2.1. To make this discussion more concrete, consider a simple example of classifying animal species (Figure 1.1) to compare an interpretable model, such as a Decision Tree (Breiman et al., 1984), with a black-box model, such as a Convolutional Neural Network (Krizhevsky et al., 2017). A Decision Tree breaks down the classification process into a series of logical rules based on features extracted from animal images, such as beak shape, wing color, and body size. These rules are represented as branches and nodes in the tree structure, making it easy to understand how the model arrived at its decision for a particular image. The decision process of the model is transparent and interpretable, allowing users to directly see the features that contribute to the classification. On the other hand, a black-box model like a Convolutional Neural Network can also be used for animal classification. Convolutional Neural Networks are powerful deep learning models capable of learning complex patterns and extracting high-level features from images.

They excel at capturing intricate details and subtle visual cues that can differentiate animal species. However, the decision process of a Convolutional Neural Networks is not directly interpretable by humans. It involves a complex network of millions of parameters organized in tens or hundreds of interconnected layers, each performing convolutions, pooling, and nonlinear transformations, which makes it challenging to understand how the model arrives at its final prediction for a specific input sample. In summary, an interpretable model, such as a Decision Tree, provides a transparent decision process with easily understandable rules, allowing users to directly interpret and trust its predictions. On the other hand, the black-box model, like a Convolutional Neural Network, can achieve high accuracy by capturing complex patterns but lacks direct interpretability. Although explanation methods can provide insights into the black-box model's decision process, they are often not faithful to the original model behavior. Therefore, the choice between the two models depends on the specific requirements of the application and the importance of interpretability in understanding and trusting the classification outcomes. The choice between an interpretable model and a black-box model depends on the specific requirements of the application, the importance of interpretability, and the acceptable level of accuracy for the task at hand. Striking a balance between accuracy and explainability remains an ongoing challenge in the field of deep learning.

1.2.2 AI fails to attain both high accuracy and good explanations

One of the key requisites for human trust is for an agent to show consistent and reliable behavior. Shen (2022) proposes to assess agents' behavior in terms of (i) task performance i.e., the capacity of the agent to provide ***accurate predictions*** for test samples, and (ii) rationale i.e., the capacity of the agent to give ***explanations*** for its predictions. Unfortunately, interpretable models usually provide high-quality explanations but may fail to solve challenging tasks. On the contrary, black-box models tend to attain high task accuracy but provide brittle and poor explanations. Ideally, instead, we would like to deploy models that attain high task performance and provide high-quality explanations at the same time. For this reason, this struggle is commonly known in the literature as the **accuracy-explainability trade-off** (Rudin, 2019).

While intense efforts lead to consistent advances in terms of explaining trained "black-box" models, most of these approaches turned out to be subject to similar limitations: they are mostly qualitative (mostly visual), local (instance-based), low-level (input-based), and post-hoc (they do not make a model trustworthy by design, they try to check if an existing model can be trusted).

A first sign of change came only recently when Koh et al. (2020) proposed to supervise the last hidden layer of neurons with human annotated concepts. This allowed the network to (i) be aware of ground-truth human concepts at training time, (ii) use learnt

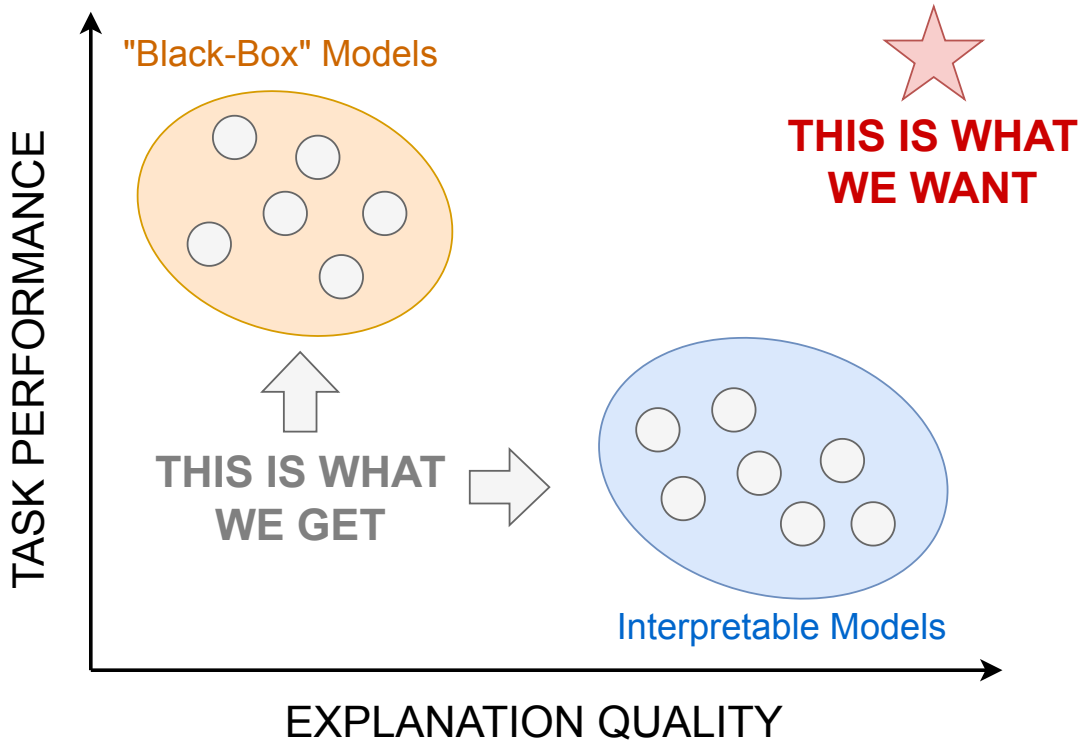


Figure 1.2: Visual representation of the accuracy-explainability trade-off. The picture shows the difference between interpretable and "black-box" (non-interpretable) models in terms of two axes: task performance and explanation quality. Interpretable models provide high-quality explanations but may fail to solve challenging tasks, while black-box models attain high task accuracy but provide brittle and poor explanations. Ideally, we would like to deploy models that attain high task performance and provide high-quality explanations at the same time.

concepts to provide more intuitive high-level explanations, and (iii) interact with human experts correcting mispredicted concepts at test time. While this design significantly improved human trust, it did not solve the whole issue as (i) the explanations were still mostly local and qualitative and (ii) enforcing concept supervisions during training lead to worse task performance (Mahinpei et al., 2021). As a result, finding a good compromise between accurate predictions and robust explanations remains one of the fundamental open problems in deep learning (see visual representation of this trade-off in Figure 1.2).

1.2.3 Explainable AI lacks a theory

A considerable number of works attempted to describe key methods and notions in this fast-growing literature (Adadi and Berrada, 2018; Das and Rad, 2020; Arrieta et al., 2020; Došilović et al., 2018; Tjoa and Guan, 2020; Gunning et al., 2019; Hoffman et al., 2018; Palacio et al., 2021). However, none of these works are grounded on a solid and unifying theory of explainability, but they rather rely on qualitative descriptions, preventing them

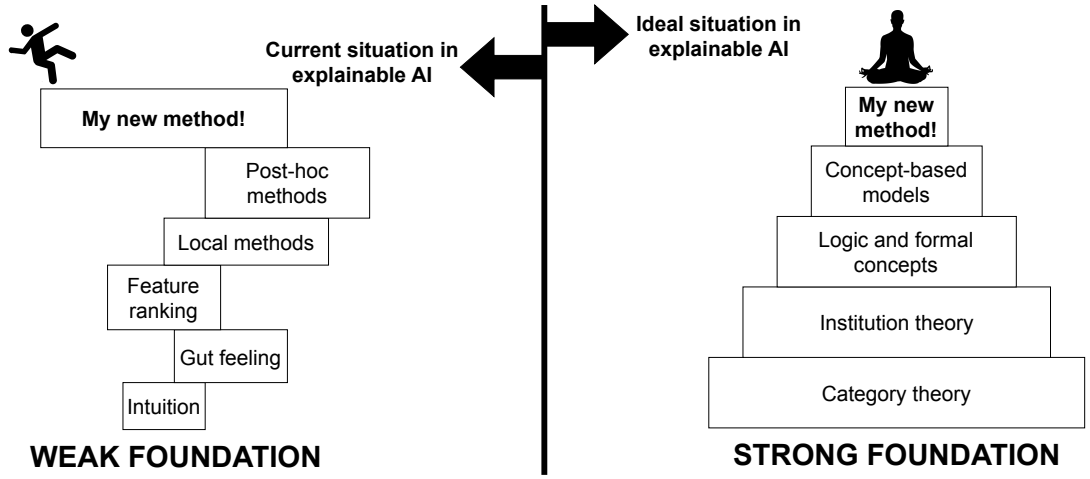


Figure 1.3: Comic representation showing two possible foundations for the explainable AI field. Explainable AI currently lacks a strong theoretical basis, which undermines the stability of the whole research field (left). A stronger theoretical foundation would enable more principled and stable development of new ideas (right).

from drawing truly universal conclusions (Figure 1.3). Current surveys acknowledge this problem and grumble that key fundamental notions of explainable AI still lack a formal definition, and that the field as a whole is missing a unifying and sound formalism (Adadi and Berrada, 2018; Palacio et al., 2021): The very notion of "*explanation*" represents a pivotal example as it still lacks a proper mathematical formalization.

As the interest for XAI methods rises inside and outside academic environments, the need for a sound formalization and encompassing taxonomy of the field grows quickly, as an essential precondition to welcome a wider audience. Indeed, the absence of a mathematical formalization of key explainable AI notions may severely undermine this research field, as it could lead to ill-posed questions, induce re-discovery of the same ideas, and make it difficult for new researchers to approach the domain.

1.3 Statement of purpose & summary of contributions

In this work I present a few initial contributions to address the knowledge gaps discussed in this chapter (see Figure 1.4 for a visual representation of the chapters' plan). To this aim I will walk backwards: starting from providing the first theoretical elements of explainable AI, up to develop interpretable deep learning methods going beyond the current accuracy-explainability trade-off. In particular this work introduces the following contributions:

- the first theory of explainable AI and concept learning formalizing some of the key notions used in this field for the first time (Chapter 2);

- logic explained networks: a novel class of concept-based models aiming to solve and explain complex tasks at the same time, without requiring external post-hoc XAI models to extract explanations (Chapter 3);
- concept embedding models: a novel class of concept-based models breaking the current accuracy-explainability trade-off and scaling to real-world conditions (Chapter 4);
- deep concept reasoners: the first differentiable concept-based model able to attain state-of-the-art performance on complex tasks while being fully interpretable (Chapter 5).

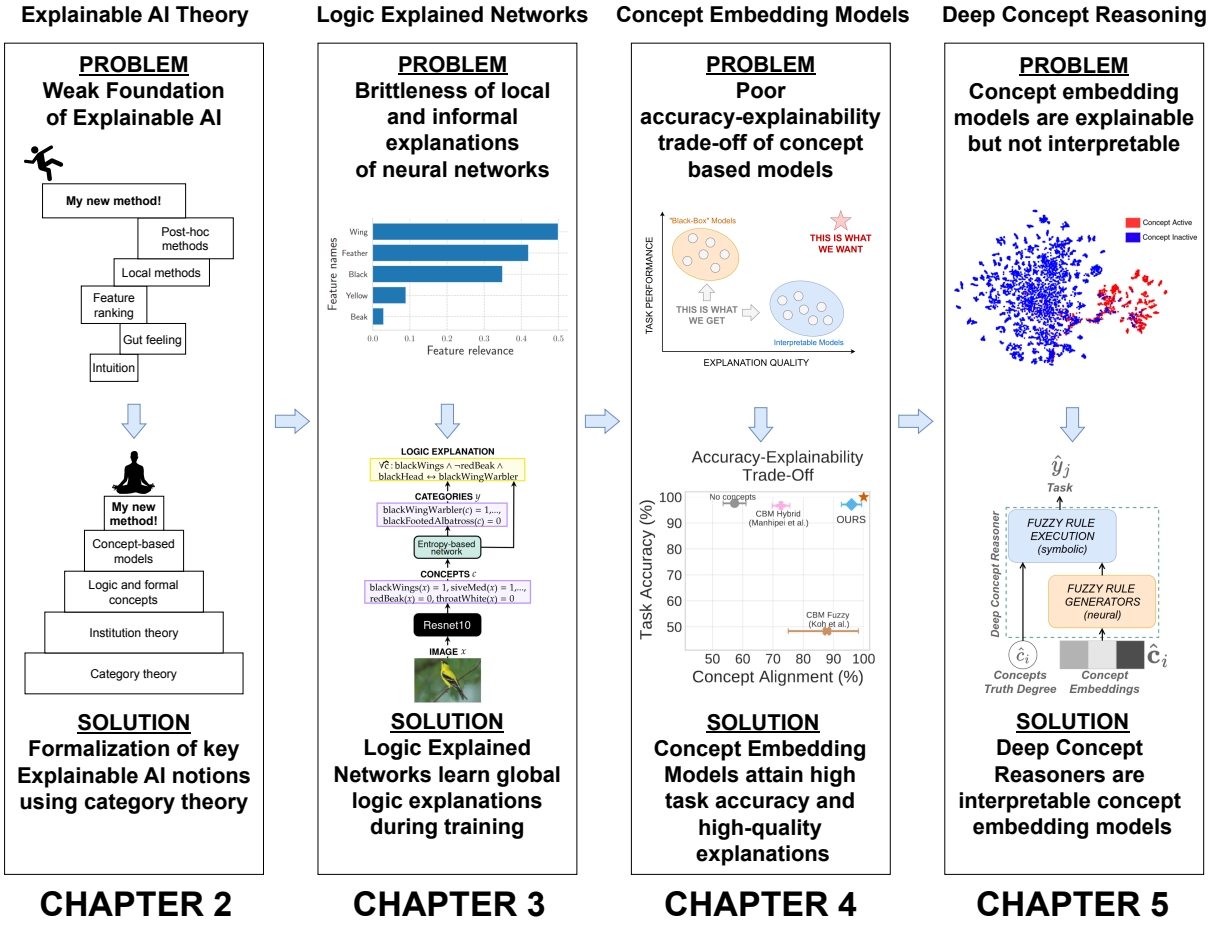


Figure 1.4: Summary of the chapter plan. The first three chapters attempt to address the three knowledge gaps outlined in the introduction (lack of formal theory for explainable AI, lack of formal explanations for deep learning models, and poor accuracy-explainability trade-off). The last chapter presents the first interpretable neural concept-based model going beyond the current accuracy-interpretability trade-off, taking the best out of Chapter 3 and Chapter 4.

1.3.1 Key outcomes

This thesis offers two essential takeaways. First, it lays the groundwork for a **foundational theory of explainable AI**. This theory provides an initial understanding of how explainable AI systems work, making their decision-making processes transparent and comprehensible to humans. Second, the thesis presents a practical framework for developing **interpretable deep learning models**. These models are designed to go ***beyond the current accuracy-interpretability trade-off***, enabling users to grasp the rationale behind deep learning-generated decisions. These two results are poised to have a significant impact on the AI community and have the potential for further development in the coming years.

The foundational theory of XAI poses a distinctive challenge, given the diverse contributions from various disciplines such as computer science, psychology, philosophy, and mathematics. All these multidisciplinary approaches and contributions enhance the richness of the field by incorporating diverse perspectives. However, the disparate backgrounds of researchers also introduce a broad spectrum of languages and logical frameworks, potentially becoming a barrier to mutual understanding. In light of these complexities, XAI requires a theoretical framework that serves two crucial functions: the formalization of concepts and the unification of the field through a language that accommodates the diverse contributions spanning different disciplines.

The framework for interpretable deep learning models offers a practical solution to the challenge of understanding and trusting deep neural networks. Deep learning has achieved remarkable success in various applications, but its "black box" nature has hindered its adoption in critical domains such as healthcare and autonomous vehicles. This framework enables the design of models that are not only highly accurate but also interpretable. It facilitates the creation of models that generate insights, providing explanations for their predictions in a way that is interpretable to humans. These models can be instrumental in fields where decision-making transparency is paramount, such as medical diagnosis and legal applications, where understanding the AI's reasoning is crucial.

1.3.2 Potential developments

Both the foundational theory of explainable AI and the framework for interpretable deep learning models provide a solid starting point for future research and innovation. The theory can be expanded and refined to accommodate the evolving landscape of AI technologies, ensuring it remains applicable and relevant. In the coming years, we can expect the development of standardized methodologies for implementing explainable AI, enhancing its usability in real-world applications. The framework for interpretable deep

learning models can be adapted to various domains, fostering a growing ecosystem of AI models that balance accuracy with transparency. With interdisciplinary collaboration, these findings can be further honed, resulting in AI systems that not only provide reliable predictions but also empower humans with valuable insights, thus fostering responsible AI development and expanding its applications across diverse sectors.

1.4 Publications

This thesis is the summary of the following works I co-authored in the past two years and half. Please refer to these works for further details on methodologies and results. I only reported in this manuscript the core ideas and key results to make this text self contained:

1. Barbiero, P., Ciravegna, G., Giannini, F., Zarlenga, M. E., Magister, L. C., Tonda, A., Lio, P., Precioso, F., Jamnik, M., and Marra, G. (2023a). Interpretable neural-symbolic concept reasoning. *arXiv preprint arXiv:2304.14068 [Accepted for publication at the International Conference of Machine Learning 2023]*

2. Barbiero, P., Fioravanti, S., Giannini, F., Tonda, A., Lio, P., and Di Lavore, E. (2023b). Categorical foundations of explainable ai: A unifying formalism of structures and semantics. *arXiv preprint arXiv:2304.14094*

3. Kazhdan, D., Dimanov, B., Magister, L. C., Barbiero, P., Jamnik, M., and Lio, P. (2023). Gci: A (g)raph (c)oncept (i)nterpretation framework. *arXiv preprint arXiv:2302.04899*

4. Zarlenga, M. E., Barbiero, P., Shams, Z., Kazhdan, D., Bhatt, U., Weller, A., and Jamnik, M. (2023). Towards robust metrics for concept representation evaluation. *arXiv preprint arXiv:2301.10367 [Accepted for publication at AAAI conference on artificial intelligence]*

5. Ciravegna, G., Barbiero, P., Giannini, F., Gori, M., Lió, P., Maggini, M., and Melacci, S. (2023). Logic explained networks. *Artificial Intelligence*, 314:103822

6. Zarlenga, M. E., Pietro, B., Gabriele, C., Giuseppe, M., Giannini, F., Diligenti, M., Zohreh, S., Frederic, P., Melacci, S., Adrian, W., et al. (2022). Concept embedding models: Beyond the accuracy-explainability trade-off. In *Advances in Neural Information Processing Systems*, volume 35, pages 21400–21413. Curran Associates, Inc

7. Jain, R., Ciravegna, G., Barbiero, P., Giannini, F., Buffelli, D., and Lió, P. (2023). Extending logic explained networks to text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*

8. Lopez-Rincon, A., Kidwai, S., Barbiero, P., Meijerman, I., Tonda, A., Lio, P., Maitland-van der Zee, A.-H., Oberski, D., Kraneveld, A., et al. (2022). A robust mrna signature obtained via recursive ensemble feature selection predicts the responsiveness of omalizumab in moderate-to-severe asthma. *Authorea Preprints*

9. Azzolin, S., Longa, A., Barbiero, P., Liò, P., and Passerini, A. (2022). Global explainability of gnns via logic combination of learned concepts. *arXiv preprint arXiv:2210.07147 [Accepted for publication at the International Conference on Learning Representations]*

10. Xuanyuan, H., Barbiero, P., Georgiev, D., Magister, L. C., and Lió, P. (2022). Global concept-based interpretability for graph neural networks via neuron analysis. *arXiv preprint arXiv:2208.10609 [Accepted for publication at AAAI conference on artificial intelligence]*

11. Magister, L. C., Barbiero, P., Kazhdna, D., Siciliano, F., Ciravegna, G., Silvestri, F., Liò, P., and Jamnik, M. (2022). Encoding concepts in graph neural networks. *Advances in neural information processing systems*. [Under review]

12. Georgiev, D., Barbiero, P., Kazhdan, D., Veličković, P., and Liò, P. (2022). Algorithmic concept-based explainable reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6685–6693

13. Barbiero, P., Ciravegna, G., Giannini, F., Lió, P., Gori, M., and Melacci, S. (2022a). Entropy-based logic explanations of neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6046–6054

14. Barbiero, P., Squillero, G., and Tonda, A. (2022b). Predictable features elimination: An unsupervised approach to feature selection. In *Machine Learning, Optimization, and Data Science: 7th International Conference, LOD 2021, Grasmere, UK, October 4–8, 2021, Revised Selected Papers, Part I*, pages 399–412. Springer

15. Barbiero, P., Torné, R. V., and Lió, P. (2021). Graph representation forecasting of patient's medical conditions: Toward a digital twin. *Frontiers in genetics*, 12

16. Barbiero, P. and Lió, P. (2020). The computational patient has diabetes and a covid. *arXiv preprint arXiv:2006.06435*

17. Deasy, J., Rocheteau, E., Kohler, K., Stubbs, D. J., Barbiero, P., Lio, P., and Ercole, A. (2020). Forecasting ultra-early intensive care strain from covid-19 in england, v1. 1.4. *MedRxiv*, pages 2020–03

Chapter 2

Categorical foundations of explainable AI and concept learning

Motivation—Explainable AI (XAI) research aims to address the human need for accurate and trustworthy AI through the design of interpretable AI models and algorithms able to explain uninterpretable AI models (Arrieta et al., 2020). Some of these methods are so effective that their impact now deeply affects other research disciplines such as medicine (Jiménez-Luna et al., 2020), physics (Schmidt and Lipson, 2009; Cranmer et al., 2019), and even pure mathematics (Davies et al., 2021).

A considerable number of works attempted to describe key methods and notions in this fast-growing literature (Adadi and Berrada, 2018; Das and Rad, 2020; Arrieta et al., 2020; Došilović et al., 2018; Tjoa and Guan, 2020; Gunning et al., 2019; Hoffman et al., 2018; Palacio et al., 2021). However, none of these works are grounded on a solid and unifying theory of explainability, but they rather rely on qualitative descriptions, preventing them from drawing truly universal conclusions. Current surveys acknowledge this problem and grumble that key fundamental notions of explainable AI still lack a formal definition, and that the field as a whole is missing a unifying and sound formalism (Adadi and Berrada, 2018; Palacio et al., 2021): The very notion of "*explanation*" represents a pivotal example as it still lacks a proper mathematical formalization. The followings represent an example of some of the best definitions currently available in literature:

> *"An explanation is an answer to a 'why?' question." Miller (2019)*
>
> *"An explanation is additional meta information, generated by an external algorithm or by the machine learning model itself, to describe the feature importance or relevance of an input instance towards a particular output classification." Das and Rad (2020)*
>
> *"An explanation is the process of describing one or more facts, such that it facilitates the understanding of aspects related to said facts (by a human consumer)."* Palacio et al. (2021).

As the interest for XAI methods rises inside and outside academic environments, the

need for a sound formalization and encompassing taxonomy of the field grows quickly, as an essential precondition to welcome a wider audience. Indeed, the absence of a mathematical formalization of key explainable AI notions may severely undermine this research field, as it could lead to ill-posed questions, induce re-discovery of the same ideas, and make it difficult for new researchers to approach the domain.

Solution—To fill this knowledge gap, in this chapter we introduce the elements of the first formal theory of explainable AI and concept learning (Barbiero et al., 2023b) aiming to:

- formalize key XAI notions for the first time;
- set the scene and motivate the work of the next chapters.

The **key innovation** of this chapter is the use of categorical structures to formalize XAI notions and processes. We use category theory as it provides a sound and abstract formalism to study general structures and systems of structures, avoiding contingent details and focusing on their very essence. For this reason, category theory represents now the standard formalism of many mathematical disciplines, including algebra (Eilenberg and MacLane, 1945), geometry (Bredon, 2012), logic (Johnstone, 2014), computer science (Goguen and Burstall, 1992), and more recently machine learning (Cruttwell et al., 2022; Ong and Veličković, 2022).

In this chapter we set the scene for the next chapters defining the key notions and presenting the notation we will follow in the rest of this work in the language of category theory. We will first discuss the basic elements of category theory analyzing the main categorical structures we will use in the following chapters (Section 2.1). We will then use such categories to formally define the syntax and the semantics of explainable AI agents and notions (Section 2.2 and 2.3). Among available semantics, we will focus on the human-friendly semantics based on concept learning (Section 2.4). Finally, we will present the main knowledge gaps in concept learning which motivate the next chapters (Section 2.5).

2.1 Elements of category theory

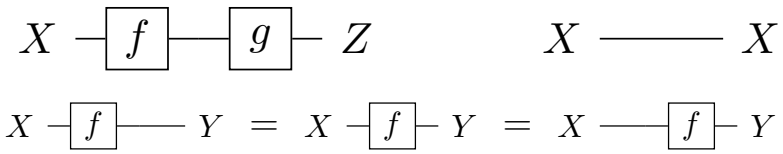
To make this work self-contained, this section introduces the minimal set of definitions that we will later need to formalize XAI systems i.e., feedback monoidal categories and the category of signatures. In particular, we will use feedback monoidal categories as a syntax to model structures sharing some of the key properties of AI systems, being able to: observe inputs, provide outputs, and receive feedback dynamically. Cartesian streams provide a semantics for these models. We will use the category of signatures to model the structure of "explanations".

2.1.1 Monoidal categories

The process interpretation of monoidal categories (Coecke and Kissinger, 2017; Fritz, 2020) sees morphisms in monoidal categories as modelling processes with multiple inputs and multiple outputs. Monoidal categories also provide an intuitive syntax for them through string diagrams (Joyal and Street, 1991). The coherence theorem for monoidal categories (Mac Lane, 1978) ensures that string diagrams are a sound and complete syntax for them and thus all coherence equations for monoidal categories correspond to continuous deformations of string diagrams. One of the main advantages of string diagrams is that they make reasoning with equational theories more intuitive.

We recall the definitions of category and monoidal category. Categories provide a syntax for processes that can be composed *sequentially*.

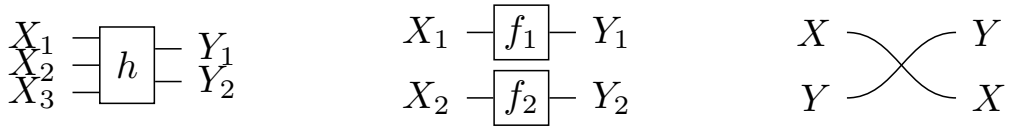
Definition 2.1.1 (Eilenberg and MacLane (1945))**.** A *category* C is given by a class of *objects* C_{obj} and, for every two objects $X, Y \in \mathsf{C}_{obj}$, a set of *morphisms* $\hom(X, Y)$ with input type X and output type Y. A morphism $f \in \hom(X, Y)$ is written $f\colon X \to Y$. For all morphisms $f\colon X \to Y$ and morphisms $g\colon Y \to Z$ there is a *composite* morphisms $f\,;g\colon X \to Z$. For each object $X \in \mathsf{C}_{obj}$ there is an *identity* morphism $\mathbb{1}_X \in \hom(X, X)$. Composition needs to be associative, i.e. there is no ambiguity in writing $f\,;g\,;h$, and unital, i.e. $f\,;\mathbb{1}_Y = f = \mathbb{1}_X\,;f$.



A mapping between two categories C_1 and C_2 that preserves compositions and identities is called a *functor* and maps objects and morphisms of C_1 into objects and morphisms of C_2.

Example 2.1.2. Set is a category whose objects are sets (e.g., $X = \{\text{tree}, \text{sky}\}$ or $Y = \{\text{green}, \text{blue}\}$) and whose morphisms are functions between sets (e.g., $f : X \to Y$ such that $\text{blue} = f(\text{sky})$ and $\text{green} = f(\text{tree})$).

Monoidal categories (Mac Lane, 1978) are categories endowed with extra structure, a monoidal product and a monoidal unit, that allows morphisms to be composed *in parallel*. The monoidal product is a functor $\otimes\colon \mathsf{C} \times \mathsf{C} \to \mathsf{C}$ that associates to two processes, $f_1\colon X_1 \to Y_1$ and $f_2\colon X_2 \to Y_2$, their parallel composition $f_1 \otimes f_2\colon X_1 \otimes X_2 \to Y_1 \otimes Y_2$. The monoidal unit is an object $I \in \mathsf{C}_{obj}$. A monoidal category is *symmetric* if there is a morphism $\sigma_{X,Y}\colon X \otimes Y \to Y \otimes X$, for any two objects X and Y, called the *symmetry*.



A symmetric monoidal structure on a category is required to satisfy some coherence conditions (Mac Lane, 1978), which ensure that string diagrams are a sound and complete syntax for symmetric monoidal categories (Joyal and Street, 1991). Like functors are mappings between categories that preserve their structure, *symmetric monoidal functors* are mappings between symmetric monoidal categories that preserve the structure and axioms of symmetric monoidal categories.

Some symmetric monoidal categories have the additional property of allowing resources and processes to be *copied* and *discarded*. These are called *Cartesian categories*. A monoidal category C is Cartesian whenever there are two morphisms, the copy $\nu_X \colon X \to X \times X$ and the discard $\epsilon_X \colon X \to 1$, that commute with all morphisms in C (Fox, 1976). When a monoidal category is Cartesian, it is customary to indicate with $\times$ the monoidal product given by the Cartesian structure, and with 1 the corresponding monoidal unit.

2.1.2 Feedback monoidal categories and Cartesian streams

With symmetric monoidal categories we can model AI systems that observe inputs and produce outputs. Most AI learning algorithms, however, rely on *feedback* to adjust their learning parameters: the learning phase for AI models is often dynamic, observing inputs, producing outputs, and getting feedback over and over again. Feedback monoidal categories provide a structure, on top of the structure of symmetric monoidal categories, to model this dynamic behaviour.

Definition 2.1.3 ((Katis et al., 2002; Di Lavore et al., 2021))**.** A feedback monoidal category is a symmetric monoidal category C endowed with an endofunctor $F : \mathsf{C} \to \mathsf{C}$, and an operation $\circlearrowleft_S \colon \hom(X \times F(S), Y \times S) \to \hom(X, Y)$ for all objects X, Y, S in C, which satisfies a set of axioms.

Feedback monoidal functors are mappings between feedback monoidal categories that preserve the structure and axioms of feedback monoidal categories. Feedback monoidal categories are the *syntax* for processes with feedback loops. When the monoidal structure of a feedback monoidal category is cartesian, we call it feedback cartesian category. Their *semantics* can be given by monoidal streams (Di Lavore et al., 2022). In cartesian

categories, these have an explicit description. We refer to them as cartesian streams, but they have appeared in the literature multiple times under the name of "stateful morphism sequences" (Sprunger and Katsumata, 2019) and "causal stream functions" (Uustalu and Vene, 2005).

Definition 2.1.4 (Uustalu and Vene (2005))**.** A *cartesian stream* $\mathbb{f}\colon \mathbb{X} \to \mathbb{Y}$, with $\mathbb{X} = (X_0, X_1, \dots)$ and $\mathbb{Y} = (Y_0, Y_1, \dots)$, is a family of functions $f_n\colon X_n \times \cdots \times X_0 \to Y_n$ indexed by natural numbers. Cartesian streams form a category $\mathsf{Stream}_{\mathsf{Set}}$.

The main purpose for using Cartesian streams is their capability of capturing an entire flow of a training process while using the framework of a category. A morphism in the category of cartesian streams encodes a process that receives an input X_n and produces an output Y_n at each time step n.

Proposition 2.1.5 (Di Lavore et al. (2022))**.** *Cartesian streams form a feedback monoidal category denoted by* $\mathsf{Stream}_{\mathsf{Set}}$.

As a result, we can use the category of Cartesian streams as semantics for structures sharing some of the key properties of AI systems, i.e. being able to: observe inputs, provide outputs, and receive feedback dynamically.

2.1.3 Free monoidal categories and syntax

A syntax is a way of reasoning abstractly about structures without the need to know the details of any given structure. In the same way a traditional syntax is defined by a set of symbols and some rules to combine them, free symmetric monoidal categories and free feedback monoidal categories are defined by a set of generators for objects and for morphisms. The rules to combine them are given by the structure and axioms of symmetric monoidal categories and feedback monoidal categories, respectively.

When reasoning with a syntax, we want to ensure that the reasoning carried out still holds in the semantics. This is done by symmetric monoidal functors, in the case of symmetric monoidal categorie, and feedback functors, in the case of feedback monoidal categories. In fact, by definition of free symmetric monoidal (resp. feedback monoidal) category, once we fix the semantics of the generators, there exist a unique symmetric monoidal (feedback monoidal) functor to the semantics category.

We will employ free feedback monoidal categories as syntax for learning agents, and take semantics in the feedback monoidal category $\mathsf{Stream}_{\mathsf{Set}}$ of cartesian streams.

2.1.4 Category of signatures

In order to model objects of type "explanation", we will use the category of signatures (Goguen and Burstall, 1992). In institution theory (Goguen and Burstall, 1992), a

Table 2.1: Reference for notation. List of main operations, objects, morphisms, categories, and functors.

Symbol	Description
	Objects
X	*Input*: the input type of a model.
Y	*Output*: the output type of a model.
P	*Parameter*: the type of a model state.
E	*Explanation*: the output type of an explainer.
	Morphisms
$\mathbb{1}_Z$	*Identity*: the identity operation on Z.
$\hat{g}$	*Model*: given an input X and parameter P, returns an output Y.
∇_Y	*Optimizer*: given a reference Y, a model output Y and a parameter P, returns the updated parameter P.
$\hat{f}$	*Explainer*: given an input X' and a parameter P, returns an output Y' and an explanation E.
	Categories
$\mathsf{Stream}_{\mathsf{Set}}$	*Cartesian streams*: feedback monoidal category of Cartesian streams on Set.
Learn	*Category of learners*: free feedback monoidal category generated by the objects X, Y, P, and the morphisms g and ∇.
XLearn	*Category of explainable learners*: free feedback monoidal category generated by the objects X, Y, P, E and the morphisms η, ∇_Y, and ∇_E.
Sign	*Category of signatures*: category generated by the object Σ and the morphism ϕ.
	Functors & Operators
$\circlearrowleft_S$	*Feedback*: the operation which brings an output back to the input.
Sen	*Sentence*: functor from the category of signatures Sign to Σ-sentences over Set.
T	*Interpreter*: functor from the category of learners Learn to Cartesian streams $\mathsf{Stream}_{\mathsf{Set}}$ over Set.

signature Σ constitutes the "syntax" of a formal language which serves as "context" or "interpretant" in the sense of classical logic (Goguen, 2005). Simple examples of signatures are given by First-Order Logic (FOL) theories and equational signatures. Signatures form a category Sign whose objects are signatures and whose morphisms $\phi : \Sigma \to \Sigma'$ are interpretations between signatures corresponding to a "change of notation" (Goguen and Burstall, 1992). From this abstract vocabulary, institution theory defines abstract statements as sentences obtained from a vocabulary Σ (Goguen and Burstall, 1992).

Definition 2.1.6 (Σ-sentence (Goguen, 2005))**.** There is a functor Sen : Sign $\to$ Set mapping each signature Σ to the set of statements $Sen(\Sigma)$. A Σ-*sentence* is an element of $Sen(\Sigma)$.

Example 2.1.7. Let Σ be a signature of propositional logic with $\{x_{flies}, x_{animal}, x_{plane}, x_{dark\_color}, \ldots\} = VAR$, being VAR an infinite set of propositional variables and the standard connectives of Boolean logic, i.e. $\neg, \wedge, \vee, \rightarrow$. Then $x_{plane} \wedge x_{flies}$ is a Σ-sentence.

2.2 Syntax and semantics of explainable AI

Here we formalize XAI structures and semantics using feedback monoidal categories and the category signatures Sign. To this end, we first formalize the notions of "learning agent" (Section 2.2.1) and "explainable learning agent" (Section 2.2.2) as morphisms in free feedback monoidal categories generated by a *model*, an *optimizer*, and an *explainer*. Then we describe a functor translating these abstract notions into concrete instances in the feedback monoidal category of $\mathsf{Stream}_{\mathsf{Set}}$ (Section 2.2.3). Finally, we formalize the notion of "explanation" as a Σ-theory and of "understanding" as a signature morphism in (Section 2.3).

2.2.1 Syntax of learning agents

Generalizing Cruttwell et al. (2022) to non-gradient-based systems, learning involves the following processes:

- **Observing** a pair of objects X and Y. In AI the objects X and Y represent input and output data of models.
- **Predicting** objects of type Y from objects of type X, given a parameters of type P.
- **Updating** parameters P according to a loss function.

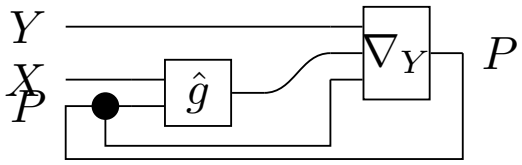
Using this informal description as guidance, we describe an abstract learning agent as a morphism in the free feedback monoidal category generated by two morphisms: a model $\hat{g}\colon X \times P \to Y$ and an optimizer $\nabla_Y\colon Y \times Y \times P \to P$. The model g produces an output of type Y given an input of type X and a parameter of type P, while ∇_Y updates the parameters of the model P given a reference of type Y, the predicted output of type Y, and the parameters P. In order to specify the syntax of abstract learning agents we define the free category Learn of abstract models and optimizers.

Definition 2.2.1 (Abstract model and optimizer)**.** The category Learn is the free feedback cartesian category generated by three objects, the input type X, the output type Y and the parameter type P, and by two morphisms, the model $\hat{g}\colon X \times P \to Y$ and the optimizer $\nabla_Y\colon Y \times Y \times P \to P$.

Remark 2.2.2. The output of the model and the reference may contain different elements, but they do have the same type, which is why we use the same type-symbol Y to represent both objects. The same argument applies in the following whenever we have conceptually different inputs/outputs but denoting objects of the same type.

Having defined the free category for abstract models and optimizers, we formalize an abstract learning agent as a morphism in Learn.

Definition 2.2.3 (Abstract learning agent)**.** An abstract learning agent is the morphism in Learn given by the composition $\circlearrowleft_P ((\mathbb{1}_Y \times \mathbb{1}_X \times \nu_P); (\mathbb{1}_Y \times \hat{g} \times \mathbb{1}_P); \nabla_Y)$:



2.2.2 Syntax of explaining agents

Compared to learning, learning to explain involves the following processes:

- **Observing** a pair of objects X and Y.
- **Predicting** objects of type Y from the observed X given a set of parameters P.
- **Predicting** explanation objects of type E from the observed X given the parameters P.
- **Updating** the parameters P according to a loss function over the predicted objects Y or E.

Remark 2.2.4. Not all XAI methods have or need to update parameters. In our formalism, we describe the updating process of these "static" systems with an identity morphism.

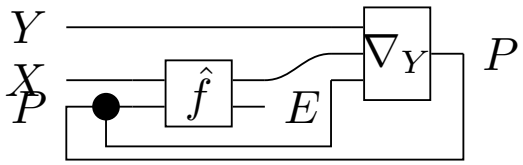
From this informal description we conclude that learning to explain extends learning processes manipulating an extra object, called "explanation". To define the category of explainable learning agents we need: (i) a new morphism $\hat{f} : X \times P \to Y \times E$, called *explainer*, that provides both predictions Y and explanations E, and (ii) a new morphism $\nabla_E : E \times E \times P \to P$ to optimize the agent parameters through its explanations E.

Definition 2.2.5 (Abstract explainer and optimizer)**.** The category XLearn is the free feedback cartesian category generated by four objects, the input type X, the output types Y and E, the parameter type P, and by three morphisms, the explainer $\hat{f} : X \times P \to Y \times E$, and the optimizer $\nabla_Y : Y \times Y \times P \to P$:

Through these new morphisms we formalize an abstract learning agent manipulating explanations.

Definition 2.2.6 (Abstract explainable learning agent)**.** An abstract explainable learning agent is the morphism in XLearn given by the composition:

$$\circlearrowleft_P \Big((\mathbb{1}_{Y\times X} \times \nu_P); (\mathbb{1}_Y \times \hat{f} \times \mathbb{1}_P); (\mathbb{1}_{Y\times Y} \times \sigma_{E,P}); (\nabla_Y \times \epsilon_E) \Big)$$



Remark 2.2.7. Our formalism allows for two different optimization processes to update the parameters of explainable learning agents. The first option optimizes explanations indirectly by applying a standard optimizer ∇_Y over the predictions of the explainer and reference labels of type Y. The second option instead optimizes explanations directly by applying the optimizer ∇_E over the explanations of the explainer and reference explanations of type E.

2.2.3 Semantics of AI agents

So far we described key notions of XAI in an abstract way, i.e. in terms of free feedback monoidal categories. However, we need these notions to be "concrete" before using them in typical AI settings. To this end, we specify the semantics of abstract agents via a functor from the free category Learn to the concrete category of Cartesian streams over Set, i.e. $\mathsf{Stream}_{\mathsf{Set}}$. With this functor we can model the specific types of inputs, outputs, parameters, explanations, and signature we need in a specific context. We refer to such a functor as "translator" and define it as follows.

Definition 2.2.8 (Translator)**.**

- A functor T_A from Learn to $\mathsf{Stream}_{\mathsf{Set}}$ is said an agent translator.
- Given a signature Σ, a functor T_Σ from XLearn to $\mathsf{Stream}_{\mathsf{Set}}$, where $\mathrm{T}_\Sigma(E) \subseteq Sen(\Sigma)$ is said an explainer translator.

Through these functors we can specify concrete semantics and describe typical AI functions and objects. For instance, we can map abstract objects and morphisms from Learn and XLearn to concrete streams of sets and functions. To simplify the notation we will use the following shortcuts: $\mathcal{X} = \mathrm{T}_A(X)$, $\mathcal{X}' = \mathrm{T}_\Sigma(X)$, $\mathcal{Y} = \mathrm{T}_A(Y)$, $\mathcal{Y}' = \mathrm{T}_\Sigma(Y)$, $\mathcal{P} = \mathrm{T}_A(P)$, $\mathcal{P}' = \mathrm{T}_\Sigma(P)$, and $\mathcal{E} = \mathrm{T}_\Sigma(E)$. In AI these objects often take values in vector spaces e.g., $\mathcal{X} \subseteq \mathbb{R}^n$, $\mathcal{P} \subseteq \mathbb{R}^p$, $\mathcal{Y} \subseteq \mathbb{R}^t$. In addition, for clarity we will denote $g = T_A(\hat{g})$,

$f = T_\Sigma(\hat{f})$ and $\hat{\nabla} = T(\nabla)$. Translator functors allow us to model different types of real-world learners, including AI agents.

Definition 2.2.9 (Concrete learning agent)**.** Given a translator T_A between Learn and $\mathsf{Stream}_{\mathsf{Set}}$, we call concrete learning agent, or simply an AI agent, the image $\mathrm{T}_A(\alpha)$ of the abstract learning agent, where $\alpha = \circlearrowleft_P ((\mathbb{1}_Y \times \mathbb{1}_X \times \nu_P); (\mathbb{1}_Y \times g \times \mathbb{1}_P); \nabla_Y)$.

We can use this formalism to provide a more precise view of the dynamic process of learning. In fact we can describe learning as the process of a concrete learning agent which keeps updating its parameters until it eventually reaches a stationary state. Given a learning agent L this process is represented by the image of L through the translator functor T_A. A learning process is convergent if there exist k such that the cartesian stream $T_A(L)$ has $g_{n+1} \approx g_n | X_n \times \cdots \times X_0$ and $X_{n+1} = X_n$ for $n > k$.

Definition 2.2.10 (Concrete explainable agent)**.** Given a translator T_Σ between XLearn and $\mathsf{Stream}_{\mathsf{Set}}$, we call concrete explainable agent, or simply an AI agent, the image $\mathrm{T}_\Sigma(\alpha)$ of the abstract explainable agent, where: $\alpha = \circlearrowleft_P \big((\mathbb{1}_{Y\times X} \times \nu_P); (\mathbb{1}_Y \times f \times \mathbb{1}_P); (\mathbb{1}_{Y\times Y} \times \sigma_{E,P}); (\nabla_Y \times \epsilon_E)\big)$.

2.3 Explanations and understanding

2.3.1 What is an explanation?

So far we just considered an "explanation" as a special object generated by an explainer morphism $\hat{f}$, depending on its input X and/or parameters P. We are now interested in analyzing the properties of this special object. As previously discussed, the XAI research community agrees in considering explanations as "answers to *why?* questions" (Miller, 2019). Here we generalize this idea providing the first formal definition of the term "explanation", which embodies the very essence and purpose of explainable AI.

Definition 2.3.1 (Explanation)**.** Given a Σ signature and a concrete explainer $f = T_\Sigma(\hat{f}) : \mathcal{X} \times \mathcal{P} \to \mathcal{Y} \times \mathcal{E}$, an explanation $\mathcal{E} = T_\Sigma(E)$ in a signature Σ is a set of Σ-sentences (i.e. a Σ-theory).

Our definition of explanation generalizes and formalizes the best definitions currently available in literature such as the ones we presented at the beginning of this chapter. In fact, existing definitions informally represent special forms of explanations. For example, according to Das and Rad (2020) and Palacio et al. (2021) an explanation provides additional meta information to describe facts related to the explainer, including the feature relevance of an input. This represent the simplest form of explanation and corresponds to a pure description of the most relevant inputs. Seminal XAI methods typically provide this form of descriptions by showing the most relevant input attributes for the prediction of a

given sample, as it happens in saliency maps (Simonyan et al., 2013), Concept Activation Vectors (Kim et al., 2018), and SHapley Additive exPlanations (Lundberg and Lee, 2017). The following example illustrates this form of explanation.

Example 2.3.2. Let Σ' be a vocabulary extending the one in Example 2.1.7 with two additional symbols $R = \{relevant, irrelevant\}$. We consider as sentences in this language, expressions of kind $(x_1 : r_1, \ldots, x_n : r_n)$, where $x_i \in VAR$ and $r_i \in R$ for $n \in \mathbb{N}$, $i = 1, \ldots, n$. Let f be an explainer aiming at predicting an output in $\mathcal{Y} = \{x_{plane}\}$ given an input in $\mathcal{X} \subseteq VAR$. Then an explanation describing the most relevant inputs is a Σ'-sentence such as $\varepsilon' = (x_{flies} : relevant, x_{animal} : relevant, x_{dark\_color} : irrelevant)$.

A more advanced form of explanation describes specific combinations of attributes leading to specific predictions. This form of explanation is common in rule-based systems such as decision trees (Breiman et al., 1984) and Generalized Additive Models (Hastie, 2017). Explanations of this form may represent an answer to a "why?" question such as to why a specific input instance leads to a specific output (Miller, 2019; Das and Rad, 2020), as we illustrate in the following example.

Example 2.3.3. Let Σ be the vocabulary in Example 2.1.7. Let f be an explainer aiming at predicting an output in $\mathcal{Y} = \{x_{plane}\}$ given an input in $\mathcal{X} \subseteq VAR$. Then the Σ-sentence $\varepsilon = x_{flies} \wedge \neg x_{animal} \rightarrow x_{plane}$ explains why the input is classified as type "plane".

Remark 2.3.4. Tarski (1944) and Goguen and Burstall (1992) proved how the semantics of "truth" is invariant under change of signature. This means that we can safely use signature morphisms to switch from one "notation" to another, inducing consistent syntactic changes in a Σ-sentence without impacting the "meaning" or the "conclusion" of the sentence (Goguen and Burstall, 1992). As a result, signature morphisms can translate a certain explanation between different signatures.

While signature morphisms do not change the meaning of an explanation, they may have a great impact on human observers as we discuss in the next section.

2.3.2 Understanding "understanding"

Tightly connected to explanation morphisms, "understanding" is another key notion in explainable AI which currently lacks a mathematical formalization. In the context of explainable AI, we are often interested in a specific type of understanding which Pritchard (2009) refers to as *understanding-why*. This form of understanding is often called *explanatory understanding* and is ascribed in sentences that take the form "I understand why Z", where Z is an explanation (for example, "I understand why the bread burnt as I left the oven on"). Using this intuition, we can formally define understanding as follows.

Definition 2.3.5 (Understanding)**.** An explainable learning agent providing explanations

in a signature Σ' can understand the explanation $\mathcal{E}$ in the signature Σ if and only if it exists at least one signature morphism $\phi : \Sigma \to \Sigma'$.

Remark 2.3.6. Notice that the existence of this morphism is not always guaranteed. This means that in some cases human observers may not be able to understand certain AI explanations. This happens even among human beings talking in two different (natural) languages. In other situations, a partial morphism may exist allowing a form of partial understanding. This happens for example in translating natural languages to formal languages.

For this reason, choosing a good signature is key and often more important for human understanding than developing state-of-the-art explainers. In fact, signatures based on ambiguous syntax (e.g., natural language) may significantly degrade human understanding as bits and pieces of explanations might get lost in the change of notation. Conversely, signatures of formal languages (e.g., propositional logic) are robust under translation in other languages, including informal languages, providing stronger guarantees for human observers. The second aspect of a good signature is the choice of the symbols providing the raw material for compound explanations. The next section illustrates how the choice of symbols plays a crucial role for human understanding.

2.4 Concepts: semantics for human understanding

2.4.1 Data semantics and human understanding

The semantics of data forms the raw material for the semantics of explanations and plays a crucial role for human understanding. We describe the semantics of data in terms of the set of attributes used to characterize each sample and on the set of values each attribute can take. We usually refer to data objects as feature and label matrixes, corresponding to input objects $\mathcal{X}$ and target $\mathcal{Y}$ respectively. The semantics of a feature matrix varies depending on the attributes which typically represent pixels in images (Kulkarni et al., 2022), relations in graphs (Li et al., 2022), words in natural language (Danilevsky et al., 2020), or semantically-meaningful variables (such as "temperature", "shape", or "color") in tabular data (Di Martino and Delmastro, 2022). Notice how different data types do not change the architecture of an explainable AI system. However, choosing a specific data type can lead to significantly different levels of human understanding. In fact, human understanding does not depend directly on the structure of the explainable AI system, but rather on the existence and completeness of a proper signature morphism from the explanation to the human observer. For example, humans lean towards explanations whose semantics is based on meaningful, human-understandable notions (such as "temperature", "shape", or "color"), rather than explanations whose semantics is based on pixels. In fact several works

show how humans do not reason in terms of low-level attributes like pixels, but rather in terms of high-level ideas (Goguen, 2005; Ghorbani et al., 2019a). Thus explanations based on such semantics might significantly improve human understanding (Ghorbani et al., 2019a). This observations have roots in cognitive sciences (e.g., Representational Theory of the Mind). According to these theories "*concepts are the basic building blocks of human thoughts*" (Margolis and Laurence, 2007): following simple rules the human mind can combine finite stocks of basic concepts over and over again to create increasingly complex representations (Margolis and Laurence, 2007). For instance, the mind can combine the basic concepts "roof" and "walls" to generate the concept "house".

2.4.2 What is a concept?

The relationship between data semantics and human understanding motivated Kim et al. (2018) to open a research line in concept learning in AI to increase human understanding. The objective of this field is to increase human trust by making AI use "the same building blocks of human thought" as opposed to other XAI approaches (Kim et al., 2018). Informally, we can define a concept as a human-understandable property shared by a set of objects. For instance "roof" is a property shared by all objects of type "house". Likewise, we can say that all objects of type "house" share the property of having a "roof". Following Ganter and Wille (1997) and Goguen (2005), we formalize the notion of "concept" as follows.

Definition 2.4.1 ((Formal) Context (Ganter and Wille, 1997))**.** A formal context $\mathcal{K} := (\mathcal{A}, \mathcal{B}, \mathcal{I})$ consists of a set of objects $\mathcal{A}$, a set of attributes $\mathcal{B}$, and a set of relations $\mathcal{I}$ between $\mathcal{A}$ and $\mathcal{B}$.

In order to express that an object $a \in \mathcal{A}$ is in relation with an attribute $b \in \mathcal{B}$, we write $(a, b) \in \mathcal{I}$ and read it as "the object a has the attribute b". For a set $\mathcal{A}' \subseteq \mathcal{A}$ of objects we can define the set of attributes common to the objects in $\mathcal{A}$: $\mathcal{A}^* := \{b \in \mathcal{B} \mid \forall a \in \mathcal{A}',\ (a, b) \in \mathcal{I}\}$. Similarly, we can define the the set of objects having all attributes in $\mathcal{B}' \subseteq \mathcal{B}$: $\mathcal{B}^* := \{b \in \mathcal{B} \mid \forall b \in \mathcal{B}',\ (a, b) \in \mathcal{I}\}$.

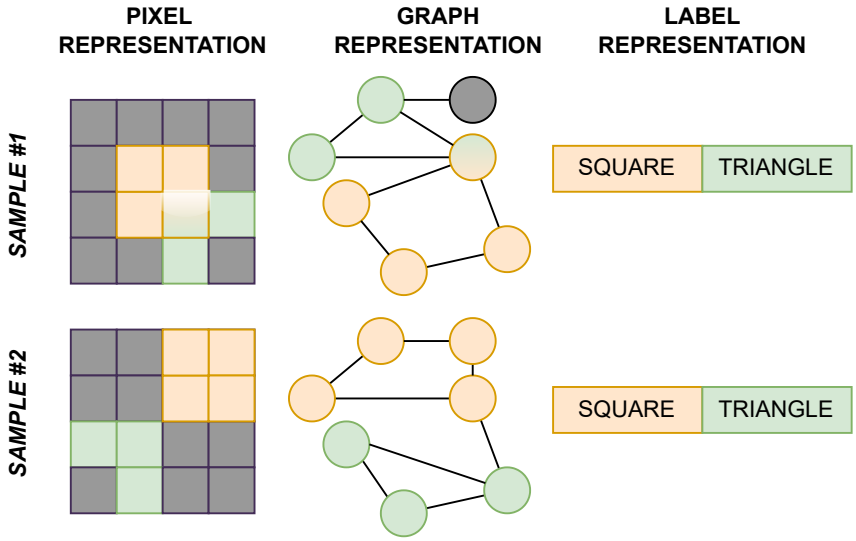
Definition 2.4.2 ((Formal) Concept (Ganter and Wille, 1997))**.** A concept of the context $\mathcal{K} := (\mathcal{A}, \mathcal{B}, \mathcal{I})$ is a pair $(\mathcal{A}', \mathcal{B}')$ such that $\mathcal{A}' \subseteq \mathcal{A}$, $\mathcal{B}' \subseteq \mathcal{B}$, $\mathcal{A}^* = \mathcal{B}$, and $\mathcal{B}^* = \mathcal{A}$.

Ganter and Wille (1997) refers to $\mathcal{A}'$ as the *extent* and to $\mathcal{B}'$ as the *intent* of the concept $(\mathcal{A}', \mathcal{B}')$. In AI, we often represent (formal) contexts using matrices where the rows are headed by sample identifiers, the columns are headed by attribute names, and the value of a cell represents the binary relation between a sample and an attribute. In these settings we often discriminate among different type of contexts depending on their use. Following common practice, we call "feature matrix" the context corresponding to

the input type of an AI agent and we represent this context with the set $\mathcal{X} \subseteq \mathbb{R}^d$. We call "label matrix" the context corresponding to the output type of an AI agent and we represent this context with the set $\mathcal{Y} \subseteq \mathbb{R}^l$.

Remark 2.4.3. The same concept can have different representations depending on its intent $\mathcal{B}'$. In particular the intent plays a key role in assigning specific semantics to the context, thus affecting the semantics of explanations and human understanding, as illustrated in the following example.

Example 2.4.4. Consider the concepts "square" and "triangle" for two samples described using different attributes i.e., pixels (image), node neighbors (graph), or labels (table):



Notice how the form of the explanations is considerably different and some are less intuitive than others. In particular notice how the two samples are slightly different when we use pixels or graphs to represent them. This means that if we use these forms of representations for explanations we may end up with slightly different explanations for each sample introducing unnecessary noise. The two labels instead are exactly the same showing how these representations can be stable and robust as they do not change for small changes in the set of samples we consider. This property makes labels suitable as the building blocks of compound explanations as explanations can rely on robust representations which do not change significantly under small perturbations.

In particular, Kim et al. (2018) observe that when the intent is less "structured" (e.g., attributes represent pixels of an image) explanations may be less coherent and less intuitive for humans. This is why Kim et al. (2018) propose to increase human understanding by providing explanations based on contexts where the individual attribute names are semantically meaningful and human-understandable (as it often happens in tabular data). For this reason, when the intent of the feature matrix is not human-understandable, Kim

et al. (2018) propose to transform the original intent into a more semantically meaningful set of attributes where concepts and explanations are more intuitive for human observers.

Remark 2.4.5. For brevity and simplicity, Kim et al. (2018) refers to human-understandable attributes as "high-level concepts" or simply as "concepts". From now on we will follow this convention and we refer to "human-understandable attributes" as "concepts" highlighting the distinction with "formal concepts" when appropriate.

The following definition will simplify our description in the next chapters.

Definition 2.4.6 (High-level concept (Kim et al., 2018))**.** A high-level concept (or simply "concept") is a human-understandable and semantically meaningful attribute name.

In the next section we describe the general structure of AI agents providing explanations in human-understandable semantics based on high-level concepts.

2.4.3 Concept-based models

Concept-based models are explainable AI agents generating predictions using human-understandable concepts as input (Kim et al., 2018; Chen et al., 2020; Koh et al., 2020). Through the input concepts, concept-based models aim to increase human trust by allowing their users to trace back predictions directly to human-understandable concepts thus making the whole decision process of the AI agent more transparent (Rudin, 2019; Shen, 2022). For instance, a concept-based model can make the prediction $\mathcal{Y} = \{x_{bird}\}$ using the concepts x_{flies} and x_{animal} allowing a human observer to verify that the set of concepts used to make the prediction matches their experience.

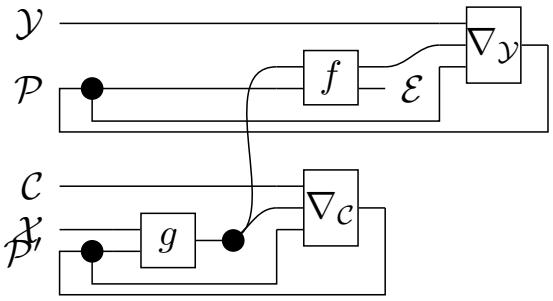
Concept-based models $f : \mathcal{C} \times \mathcal{P} \to \mathcal{Y}$ learn a map from a set of semantically meaningful concepts $\mathcal{C}$ to a set of tasks $\mathcal{Y}$ (Yeh et al., 2020). This way humans can interpret this mapping by tracing back predictions to the most relevant concepts (Ghorbani et al., 2019a). When the features of the input space are hard for humans to reason about (such as pixel intensities), we may still apply concept-based models on the output of a "concept-encoder" i.e., a mapping $g : \mathcal{X} \times \mathcal{P}' \to \mathcal{C}$ from the input space $\mathcal{X}$ to the concept space $\mathcal{C}$ (Ghorbani et al., 2019b; Koh et al., 2020). Using our categorical constructions we can formally describe a concept-based model as follows.

Definition 2.4.7 (Concept encoder)**.** A concept encoder is an AI agent $g : \mathcal{X} \times \mathcal{P}' \to \mathcal{C}$ where the output object $\mathcal{C}$ represents a set of concepts[1].

Definition 2.4.8 (Concept-based model)**.** Given a concept encoder $g : \mathcal{X} \times \mathcal{P}' \to \mathcal{C}$, a concept-based model is a XAI model where the explainer $f : \mathcal{C} \times \mathcal{P} \to \mathcal{Y} \times \mathcal{E}$ takes as

[1]Concepts in the sense of Kim et al. (2018).

input object the set of concepts $\mathcal{C}$ generated by the concept encoder:



Training a concept-based model may require a dataset where each sample consists of input features $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ (e.g., an image's pixels), k ground truth concepts $\mathbf{c} \in \mathcal{C} \subseteq \{0,1\}^k$ (i.e., a binary vector with concept annotations, when available) and t task labels $\mathbf{y} \in \mathcal{Y} \subseteq \{0,1\}^t$ (e.g., an image's classes). During training, a concept-based model is encouraged to align its predictions to task labels i.e., $\mathbf{y} \approx \hat{\mathbf{y}} = f(g(\mathbf{x}))$. Similarly, a concept encoder can be supervised when concept labels are available i.e., $\mathbf{c} \approx \hat{\mathbf{c}} = g(\mathbf{x})$. When concept labels are not available, unsupervised concept encoders extract concepts by associating concept labels to clusters found in the embeddings of pre-trained models as proposed by Ghorbani et al. (2019b); Magister et al. (2021). We indicate concept and task predictions as $\hat{c}_i = (g(\mathbf{x}))_i$ and $\hat{y}_j = (f(\hat{\mathbf{c}}))_j$ respectively.

Remark 2.4.9. In the following chapters we will omit the dependency on parameters for morphisms as they are all parametric i.e., instead of writing $f : \mathcal{C} \times \mathcal{P} \to \mathcal{Y}$ we will simply write $f : \mathcal{C} \to \mathcal{Y}$.

2.5 Knowledge gaps and aims

We can summarize the ultimate aim of XAI research on concepts as follows: To design trustworthy AI systems able to attain state-of-the-art performance in solving complex tasks while providing human-understandable explanations for their decisions. To this end, XAI research on concepts focuses on four main research areas: models, representations, metrics, and explanations. Research in concept models aims to improve the architectures of concept-based models and their concept encoders to increase the performance of these models in learning concepts from raw features and in learning the task labels from the learnt concepts. Research in concept representations focuses on devising more efficient data structures to encapsulate the information of learnt concepts preserving their semantics but allowing for concept encoders to incorporate sample-specific information about specific concept instances. Research in concept metrics aims to assess the quality of learnt concepts in terms of preserved semantics and their predictive information for task labels. Finally, research in concept explanations targets the design of signatures and the forms of the

explanations provided by concept-based models in order to make them more trustworthy.

However, XAI research on concepts is a relatively young field and current approaches represent only the first steps towards the ultimate goal of the field. In fact, current approaches struggle either to attain state-of-the-art performances in solving complex tasks or to preserve a clean semantics in learnt concept representations. In addition, state-of-the-art concept-based systems either provide simple explanations in non-formal languages (e.g., Concept Activation Vectors (Kim et al., 2018) or Concept Bottleneck Models (Koh et al., 2020)), which may mislead human observers, or are not differentiable thus impeding a joint training with concept encoders to learn better concepts depending on the task (e.g., decision trees (Breiman et al., 1984) or Bayesian rule lists (Letham et al., 2015)). We can then summarize the main research directions in this field as follows:

Aim #1 — Generate compound explanations in formal languages with differentiable concept-based models;

Aim #2 — Attain state-of-the-art performance in solving complex tasks while preserving clean concept semantics;

The following chapters address some of the main knowledge gaps currently arising in different areas of XAI concept research. In particular, Chapter 3 focuses on **Aim #1** presenting Logic Explained Networks (LENs), a family of differentiable concept-based models generating compound explanations in the formal language of first-order logic. Chapter 4 focuses on **Aim #2** introducing concept embedding representations which allow concept-based models to attain state-of-the-art performance in solving complex tasks while preserving clean concept semantics. While addressing **Aim #2**, existing concept-based models are not designed for concept embeddings and are unable to provide formal and semantically meaningful explanations based on this concept representation. To solve this limitation, Chapter 5 presents the Deep Concept Reasoner (DCR), the first interpretable concept-based model using concept embeddings. In particular DCR represents the first differentiable concept-based model attaining state-of-the-art performance in solving complex tasks while providing human-understandable and formal explanations for its decisions, thus representing a concrete step towards efficient and trustworthy AI systems.

Chapter 3

Logic explanations of neural networks (beyond feature ranking)

Motivation—In the previous chapter we discussed the main knowledge gaps in the explainable AI literature we aim to address in this work. In particular, we discussed the limits of state-of-the-art concept-based models in providing formal explanations with differentiable architectures. On the one hand, while several concept-based models (such as decision trees) provide formal logic statements, they are not differentiable thus preventing the task loss to update and improve the concept encoder's parameters. On the other hand, current differentiable concept-based models are limited to simple, local, or informal explanations. Logistic regression represents a notable example of such models as it can only provide linear explanations (e.g., it cannot even solve simple task such as mutual exclusivity of concepts) and provides explanations in terms of feature importances (associated to the feature weights of the model). As discussed in the previous chapter, feature importances represent a key step of explainability as they identify the key elements of an explanation. However, feature importances alone may not be enough as they do not illustrate the reasoning steps required to solve the task. For these reasons concept-based models are currently limited to solve and explain simple tasks or to be detached from concept encoders.

Solution—To fill this gap, in this chapter we present (entropy-based) Logic Explained Networks (LENs, (Barbiero et al., 2022a; Ciravegna et al., 2023)), a novel class of concept-based models aiming to:

- provide compound formal explanations illustrating the key concepts required to solve a task and how these concepts are combined in the decision process;
- solve and explain complex tasks at the same time, without requiring external post-hoc XAI models to extract explanations;
- allow the gradient of the task loss to flow back and update the parameters of the concept encoder.

The **key innovation** of LENs is a sparse attention layer to select the key concepts in neural concept-based models. The sparse attention allows the model to learn how to cherry-pick the most relevant concepts for each task and use only them for solving the task. The selection of few concepts allows the extraction of simple logic explanations as LENs work between two semantically meaningful spaces i.e., the concept and the task space. The generation of explanations in the formal language of logic also enables more quantitative evaluations of explanations in terms of prediction accuracy and explanation complexity.

In this chapter we will discuss the advantages of providing logic explanations for neural networks (Section 3.1). We will then present the general framework of Logic Explained Networks i.e., neural concept-based models providing first-order logic explanations (Section 3.2), and among these models we will present the details of the entropy-based Logic Explained Network (Section 3.3). The remaining of the chapter: (i) describes the key metrics to evaluate Logic Explained Networks (Section 3.4), (ii) illustrates the experimental setup we use to benchmark these models (Section 3.5), (iii) discusses the results of the experiments (Section 3.6), and (iv) summarizes the key findings and limitations of the proposed approach (Section 3.7).

3.1 Why logic explanations?

A logic explanation $\varphi \in \mathcal{E}$ can be considered a special kind of a concept-based explanation, where the description is given in terms of logic predicates, connectives and quantifiers. This set of terms determines the specific logic signature Σ used to generate explanations. For instance, an explanation in the first-order logic (FOL) signature $\Sigma(FOL)$ may look like: "$\forall x : \ \mathit{is\_human}(x) \rightarrow \mathit{has\_hands}(x) \wedge \mathit{has\_head}(x)$", that reads "being human implies having hands and head". Here, the concepts are "human", "hands", and "head", and the logic sentence is human-interpretable explanation of a pattern. In general, concept-based models aid human-understanding as they learn mappings from two semantically meaningful sets of symbolic attributes e.g., concepts and tasks. However, compared to other concept-based techniques, logic-based explanations provide many key advantages, that we briefly describe in what follows. A logic explanation reported is a rigorous and unambiguous statement (clarity). This formal clarity may serve cognitive-behavioral purposes such as engendering trust, aiding bias identification, or taking actions/decisions. For instance, dropping quantifiers and variables for simplicity, the formula "$\mathit{snow} \wedge \mathit{tree} \leftrightarrow \mathit{wolf}$" may easily outline the presence of a bias in the collection of training data. Different logic-based explanations can be combined to describe groups of observations or global phenomena (modularity). For instance, for an image showing only the face of a person, an explanation could be "$(\mathit{nose} \wedge \mathit{lips}) \rightarrow \mathit{human}$", while for another image showing a person from behind

a valid explanation could be "(*feet* $\wedge$ *hair* $\wedge$ *ears*) $\rightarrow$ *human*". The two local explanations can be combined into "(*nose* $\wedge$ *lips*) $\vee$ (*feet* $\wedge$ *hair* $\wedge$ *ears*) $\rightarrow$ *human*". The quality of logic-based explanations can be quantitatively measured to check their correctness and completeness (measurability). For instance, once the explanation "(*nose* $\wedge$ *lips*) $\vee$ (*feet* $\wedge$ *hair* $\wedge$ *ears*)" is extracted for the class *human*, this logic formula can be applied on a test set to check its generality in terms of quantitative metrics like accuracy, fidelity and consistency. Further, logic explanations can be rewritten in different equivalent forms such as in *Disjunctive Normal Form* (DNF) and *Conjunctive Normal Form* (CNF) (versatility). Finally, techniques such as the Quine–McCluskey algorithm can be used to compact and simplify logic explanations (McColl, 1878; Quine, 1952; McCluskey, 1956) (simplifiability). As a toy example, consider the explanation "(*person* $\wedge$ *nose*) $\vee$ ($\neg$*person* $\wedge$ *nose*)", that can be easily simplified in "*nose*". For all these reasons, logic-based formulae represent a robust and sound form of explanations.

3.2 Logic explained networks

Here we present a novel family of neural models, the *Logic Explained Networks* (LENs), which are trained to *solve-and-explain* a categorical learning problem integrating elements from deep learning and logic. Differently from vanilla neural architectures, LENs can be directly interpreted by means of a set of logic formulae. In order to implement such a property, LENs require their inputs and outputs to represent the activation scores of human-understandable symbols (i.e., meaningful concepts/tasks). Then, specifically designed learning objectives allow LENs to make predictions in a way that is well suited for providing formal explanations that involve the input concepts. In order to reach this goal, LENs leverage parsimony criteria aimed at keeping their structure simple.

Formally, a LEN is a mapping $f : \mathcal{C} \rightarrow \mathcal{Y}$, where $\mathcal{C} = [0, 1]^k$ is the *input concept space*, and $\mathcal{Y} = [0, 1]^t$ is the *task or output concept space*, as described in the previous chapter. Logic explanations produced by a LEN describe the relationships between the tasks and the input concepts. In particular, a LEN output corresponding to the i-th task (i.e., f_i) can be directly translated into a logic rule $\varphi_i \in \mathcal{E}$ that involves the input concepts.

This section presents the fundamental methods used to implement Logic Explained Networks. We start by describing the procedure that is used to extract logic rules out of LENs for an individual observation or a group of samples (Section 3.2.1). We then discuss how to constrain LENs to yield concise logic formulae using ad-hoc parsimony criteria (Section 3.3) to bound the complexity of explanations.

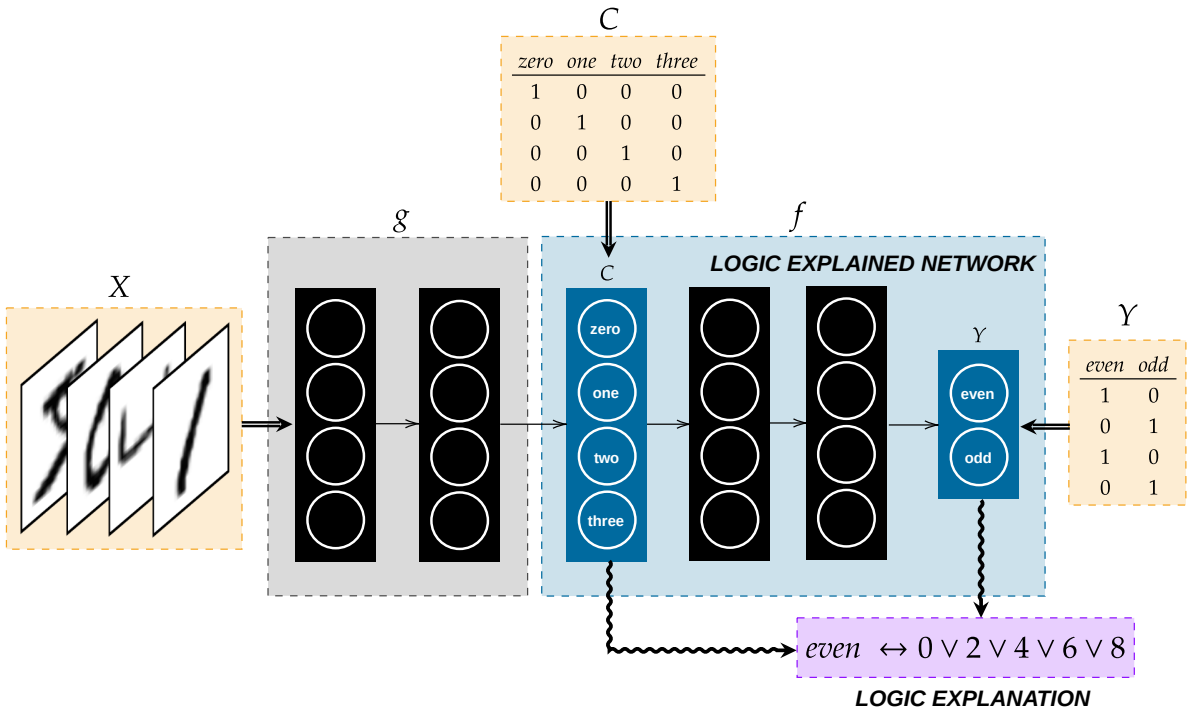


Figure 3.1: Logic Explained Network (LEN). The LEN is placed on top of a concept-encoder g which maps the input data into a first set of interpretable concepts. The figure shows a MNIST example: Handwritten digits are first classified by g. A LEN is then employed to classify and explain whether the digit is even or odd.

3.2.1 Neural networks and logic explanations

To allow the extraction of logic formulae, any LEN $f = (f_1, \ldots, f_t)$ requires both its inputs $\mathcal{C}$ and outputs $\mathcal{Y}$ to belong to the real-unit interval which allows us to interpret LENs as logic maps. This way, given a LEN, a logic formula can be associated to each output task f_i. As it will become clear shortly, we extract logic formulae from LENs by inspecting the learnt input/output maps.

We have already introduced the notation φ_i to indicate the logic explanation of the output concept i. This generic notation will be properly formalized in the following. We overload the symbol φ_i to explicitly indicate, when needed, the data subset where the logic explanation holds true, using the notation $\varphi_{i,\cdot}$. Here the second subscript can refer either to a single data sample c, $\varphi_{i,c}$, or to a set $\mathcal{S}$ of data samples, $\varphi_{i,\mathcal{S}}$. In practice, $\mathcal{S}$ denotes the region of the concept space that is covered by the i-th explanation, i.e. the set of concept tuples for which the formula $\varphi_{i,\mathcal{S}}$ is true. By aggregating over multiple samples, the scope of the logic formula may be tuned from strictly local example-level explanations ($\mathcal{S} = \{c\}$) to set-level explanations ($\mathcal{S} \subseteq \mathcal{C}$), where the latter can be focused on a precise class, i.e., class-level explanations. Eventually, for $\mathcal{S} = \mathcal{C}$, global logic formulae holding on the whole concept space $\mathcal{C}$ can be extracted.

The process of extracting logic explanations begins with a forward pass when f maps

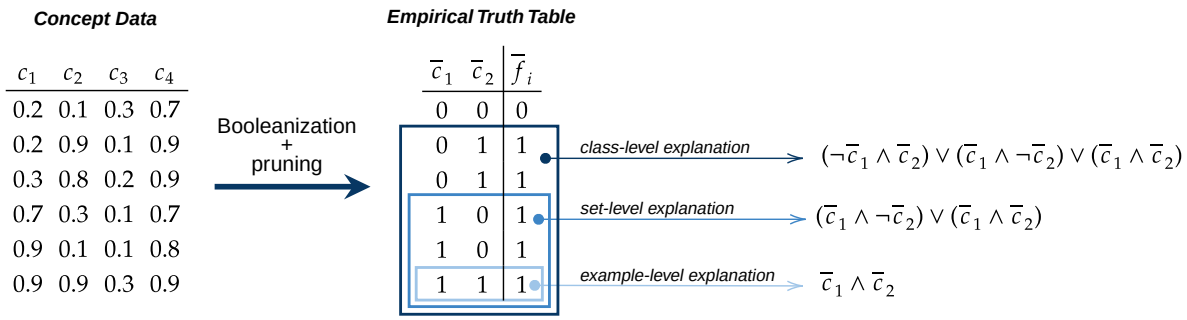


Figure 3.2: Empirical truth table $\mathcal{T}^i$ of the i-th LEN output f_i, with $k = 4$ concepts and $m_i = 2$ relevant concepts. Concept Booleanization yields the same example-level explanations for similar samples, thus simplifying task-level explanations.

the samples in $\mathcal{C}$ into the task space $\mathcal{Y}$. After this forward pass, both the input data $\mathcal{C}$ and the predictions of f are thresholded with respect to a reference value (e.g., 0.5) to obtain their Boolean interpretation. Then, for each output neuron i, an *empirical truth-table* $\mathcal{T}^i$ is built by concatenating the k-columns of Booleanized input concept tuples $\{\bar{c} : \ c \in \mathcal{C}\}$, with the column of the corresponding LEN's predictions $\bar{f}_i(c)$ (left-side of Fig. 3.2). The truth-table $\mathcal{T}^i$ can be converted into a logic formula φ_i in Disjunctive Normal Form (DNF) as commonly done in related literature (Mendelson, 2009). However, the rationale behind LENs is to extract formulae that are simple, emphasizing the most relevant relationships among the input concepts, according to specific parsimony criteria (that will be the subject of Section 3.3). Thus, any f_i will depend only on a proper subset of $m_i \leq k$ concepts, and the formula φ_i will be built according to the restriction of $\mathcal{T}^i$ to $m_i \leq k$ columns (see e.g. Fig. 3.2). Notice that, for convenience in the notation, we assumed the first m_i columns to be the ones playing a role in the explanation, even if they could be any set of m_i columns of $\mathcal{T}^i$.

3.2.2 Example-level explanations

In order to give more details about the rule extraction, we formally introduce the set $\mathcal{O}_i = \{c \in \mathcal{C} : \ \bar{f}_i(c) = 1\}$ as the set of all the sampled concept tuples that make true the i-th output explanation, i.e. the *support* of $\bar{f}_i$. Given a sample $c \in \mathcal{O}_i \subseteq \mathcal{C}$, the Booleanization $\bar{c}$ of its continuous features may provide a natural way to get an example-level logic explanation $\varphi_{i,c}$. To make logic formulae more interpretable, the notation $\tilde{c}$ denotes human-interpretable strings representing the concept names or their negation,

$$\varphi_{i,c} = \tilde{c}_1 \wedge \ldots \wedge \tilde{c}_{m_i} \qquad \text{where } \tilde{c}_j := \begin{cases} \bar{c}_j, & \text{if } c_j \geq 0.5 \\ \neg\bar{c}_j, & \text{if } c_j < 0.5 \end{cases}, \text{ for } j = 1, \ldots, m_i \tag{3.1}$$

3.2.3 Set-level and task-level explanations

By considering Eq. 3.1 for all $c \in \mathcal{S}$, with $\mathcal{S} \subseteq \mathcal{O}_i$, and aggregating all the example-level explanations, an explanation for a set of samples can be generated as follows:

$$\varphi_{i,\mathcal{S}} = \bigvee_{c \in \mathcal{S}} \varphi_{i,c} = \bigvee_{c \in \mathcal{S}} \tilde{c}_1 \wedge \ldots \wedge \tilde{c}_{m_i} \tag{3.2}$$

As some formulae $\varphi_{i,c}$ might be equivalent for different samples c, duplicated instances can be discarded keeping only one of them, without loss of generality. In case $\mathcal{S} = \mathcal{O}_i$, we simply write φ_i in place of $\varphi_{i,\mathcal{S}}$ and we refer to such set-level explanation as the task-level explanation corresponding to the i-th LEN output i.e., f_i.

Example 3.2.1. Consider the Boolean XOR function, defined by $xor(0,0) = xor(1,1) = 0$, $xor(1,0) = xor(0,1) = 1$. Let $f = [f_1]$ be a LEN that has been trained to approximate the XOR function. Considering the two samples $c^1 = (0.2, 0.7)$, $c^2 = (0.6, 0.3)$, their Boolean representations are $\bar{c}^1 = (0,1)$, $\bar{c}^2 = (1,0)$, and therefore $\bar{f}_1(c^1) = \bar{f}_1(c^2) = 1$. These examples yield the example-level explanations $\varphi_{1,c^1} = \neg\bar{c}_1 \wedge \bar{c}_2$ and $\varphi_{1,c^2} = \bar{c}_1 \wedge \neg\bar{c}_2$ respectively. As a result, the set-level explanation for f_1 is given by $\varphi_1 = (\neg\bar{c}_1 \wedge \bar{c}_2) \vee (\bar{c}_1 \wedge \neg\bar{c}_2)$, which correctly matches the truth-table of the Boolean XOR function.

The methodologies described so far illustrate how logic-based explanations can be aggregated to produce a wide range of explanations, from the characterization of individual observations to formulae explaining model predictions for all the samples leading to the same output concept activation. The formula for a whole class can be obtained by aggregating all the minterms corresponding to example-level explanations of all the observations having the same concept output. In theory, this procedure may lead to overly long formulae as each minterm may increase the complexity of the explanation. In practice, we observe that many observations share the same logic explanation, hence their aggregation may not change the complexity of the task-level formula (right-side Fig. 3.2). In general, in Herbert Simon's words, "*satisficing*" class-level explanations can be generated by aggregating the most frequent explanations for each task, avoiding a sort of "explanation overfitting" with the inclusion of noisy minterms which may correspond to outliers (Simon, 1956). The criterion we followed to prune noisy terms consists in including in a global explanation only the local formulae that increase the accuracy of the explanation when measured on a validation set, sorting formulae by their support (largest support first).

A possible limitation of the described methods can be the readability of logic rules. This may occur when (i) the number of input concepts (the length of any minterm) $k \gg 1$, or (ii) the size of the support $|\mathcal{O}_i| \gg 1$ (possibly getting too many different minterms for any f_i). The greater k is, the more different concepts are available for example-level explanations. As a consequence, it will be less probable to find common

minterms for different examples and aggregate them into concise class-level explanations. In these scenarios, viable approaches to generate shorter logic rules are needed to provide interpretable explanations. We discuss how to solve these limitations and generate concise explanations in the next section.

3.3 Entropy-based logic explained networks

When humans compare a set of explanations outlining the same outcomes, they tend to have an implicit bias towards the simplest one, as outlined in philosophy (Soklakov, 2002; Rathmanner and Hutter, 2011), psychology (Miller, 1956; Cowan, 2001), and decision making (Simon, 1956, 1957, 1979). Over the years, researchers have proposed many approaches to integrate "*the law of parsimony*" into learning machines to make models more robust and to extract simpler explanations. For instance, Bayesian priors (Wilson, 2020) and weight regularization (Kukačka et al., 2017) are two of the most famous techniques to instantiate the Occam's razor principle in statistics and machine learning. In the case of LENs, the notion of simplicity is implemented by encouraging each task to depend on the smallest number of input concepts.

The proposed entropy-based approach encodes this inductive bias in an end-to-end differentiable model. The purpose of the entropy-based linear layer is to encourage the neural model to pick a limited subset of input concepts, allowing it to provide concise explanations of its predictions. The learnable parameters of the layer are the usual weight matrix W and bias vector b. In the following, the forward pass is described by the operations going from Eq. 3.3 to Eq. 3.6 while the generation of the truth tables from which explanations are extracted is formalized by Eq. 3.7. We describe the forward pass from the point of view of a single task predicted by the i-th LEN output f^i.

3.3.1 Selection of relevant concepts

The relevance of each input concept can be summarized in a first approximation by a measure that depends on the values of the weights forwarding such concept to the next layers. Considering the j-th input concept, we indicate with W_j^i the vector of weights departing from the j-th input (see Fig. 3.3), and we introduce

$$\gamma_j^i = ||W_j^i||_1 \ . \tag{3.3}$$

The higher γ_j^i, the higher the relevance of the concept j for the network f^i. In the limit case ($\gamma_j^i \to 0$) the model f^i drops the j-th concept out. To select only few relevant concepts for each target class, concepts are set up to compete against each other. To this aim, the relative importance of each concept to the i-th class is summarized in the categorical

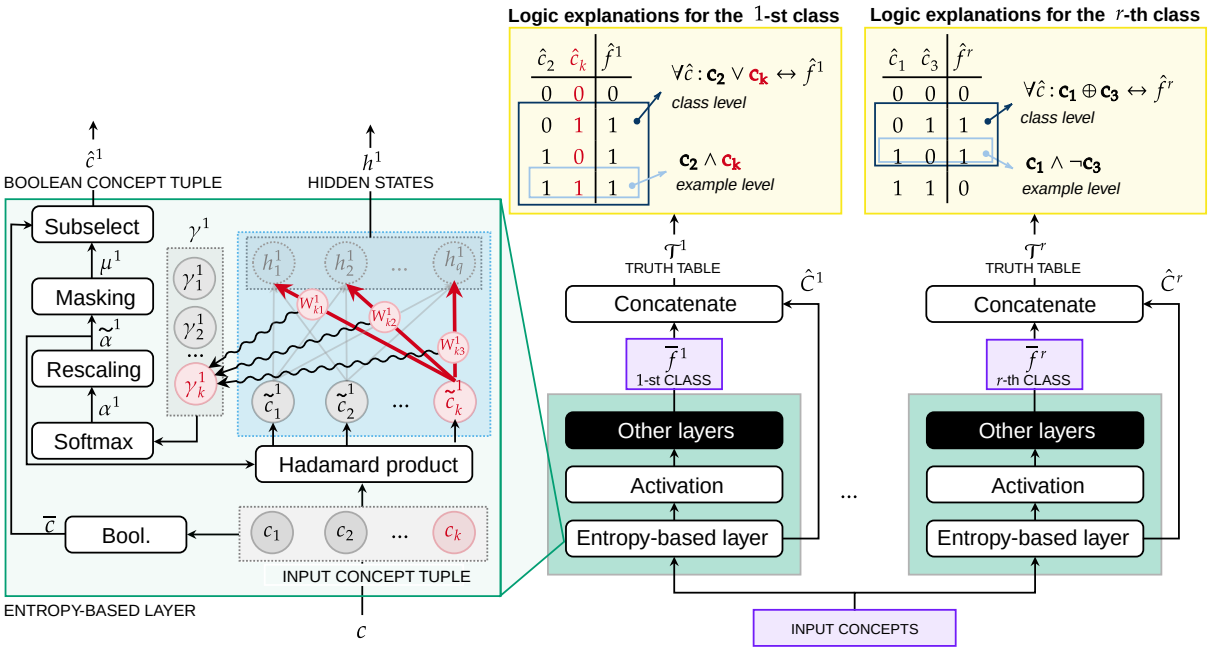


Figure 3.3: On the right, the proposed neural network learns the function $f : \mathcal{C} \rightarrow \mathcal{Y}$. For each class, the network leverages one "head" of the entropy-based linear layer (green) as first layer. For each target class i, the network provides: the class membership predictions f^i and the truth table $\mathcal{T}^i$ (Eq. 3.8) to distill FOL explanations (yellows, top). On the left, a detailed view on the entropy-based linear layer for the 1-st class, emphasizing the role of the k-th input concept as example: (i) the scalar γ_k^1 (Eq. 3.3) is computed from the set of weights connecting the k-th input concept to the output neurons of the entropy-based layer; (ii) the relative importance of each concept is summarized by the categorical distribution α^1 (Eq. 3.4); (iii) rescaled relevance scores $\tilde{\alpha}^1$ drop irrelevant input concepts out (Eq. 3.5); (iv) hidden states h^1 (Eq. 3.6) and Boolean-like concepts $\hat{c}^1$ (Eq. 3.7) are provided as outputs of the entropy-based layer.

distribution α^i, composed of coefficients $\alpha_j^i \in [0, 1]$ (with $\sum_j \alpha_j^i = 1$), modeled by the softmax function:

$$\alpha_j^i = \frac{e^{\gamma_j^i/\tau}}{\sum_{l=1}^{k} e^{\gamma_l^i/\tau}} \tag{3.4}$$

where $\tau \in \mathbb{R}^+$ is a user-defined temperature parameter to tune the softmax function. For a given set of γ_j^i, when using high temperature values ($\tau \rightarrow \infty$) all concepts have nearly the same relevance. For low temperatures values ($\tau \rightarrow 0$), the probability of the most relevant concept tends to $\alpha_j^i \approx 1$, while it becomes $\alpha_k^i \approx 0$, $k \neq j$, for all other concepts. As the probability distribution α^i highlights the most relevant concepts, this information is directly fed back to the input, weighting concepts by the estimated importance. To avoid numerical cancellation due to values in α^i close to zero, especially when the input dimensionality is large, we replace α^i with its normalized instance $\tilde{\alpha}^i$, still in $[0, 1]^k$, and

each input sample $c \in \mathcal{C}$ is modulated by this estimated importance,

$$\tilde{c}^i = c \odot \tilde{\alpha}^i \qquad \text{with} \qquad \tilde{\alpha}^i_j = \frac{\alpha^i_j}{\max_u \alpha^i_u}, \tag{3.5}$$

where $\odot$ denotes the Hadamard (element-wise) product. The highest value in $\tilde{\alpha}^i$ is always 1 (i.e. $\max_j \tilde{\alpha}^i_j = 1$) and it corresponds to the most relevant concept. The embeddings h^i are computed as in any linear layer by means of the affine transformation:

$$h^i = W^i \tilde{c}^i + b^i. \tag{3.6}$$

Whenever $\tilde{\alpha}^i_j \to 0$, the input $\tilde{c}^i_j \to 0$. This means that the corresponding concept tends to be dropped out and the network f^i will learn to predict the i-th class without relying on the j-th concept.

In order to get logic explanations, the proposed linear layer generates the truth table $\mathcal{T}^i$ formally representing the behaviour of the neural network in terms of Boolean-like representations of the input concepts. In detail, we indicate with $\bar{c}$ the Boolean interpretation of the input tuple $c \in \mathcal{C}$, while $\mu^i \in \{0,1\}^k$ is the binary mask associated to $\tilde{\alpha}^i$. To encode the inductive human bias towards simple explanations (Miller, 1956; Cowan, 2001; Ma et al., 2014), the mask μ^i is used to generate the binary concept tuple $\hat{c}^i$, dropping the least relevant concepts out of c,

$$\hat{c}^i = \xi(\bar{c}, \mu^i) \quad \text{with} \quad \mu^i = \mathbb{1}_{\tilde{\alpha}^i \geq \epsilon} \quad \text{and} \quad \bar{c} = \mathbb{1}_{c \geq \epsilon}, \tag{3.7}$$

where $\mathbb{1}_{z \geq \epsilon}$ denotes the indicator function that is 1 for all the components of vector z being $\geq \epsilon$ and 0 otherwise (considering the unbiased case, we set $\epsilon = 0.5$). The function ξ returns the vector with the components of $\bar{c}$ that correspond to 1's in μ^i (i.e. it sub-selects the data in $\bar{c}$). As a results, $\hat{c}^i$ belongs to a space $\hat{\mathcal{C}}^i$ of m_i Boolean features, with $m_i < k$ due to the effects of the subselection procedure.

3.3.2 Generation of truth-tables

The truth table $\mathcal{T}^i$ is a particular way of representing the behaviour of network f^i based on the outcomes of processing multiple input samples collected in a generic dataset $\mathcal{C}$. As the truth table involves Boolean data, we denote with $\hat{\mathcal{C}}^i$ the set with the Boolean-like representations of the samples in $\mathcal{C}$ computed by ξ, Eq. 3.7. We also introduce $\bar{f}^i(c)$ as the Boolean-like representation of the network output, $\bar{f}^i(c) = \mathbb{1}_{f^i(c) \geq \epsilon}$. The truth table $\mathcal{T}^i$ is obtained by stacking data of $\hat{\mathcal{C}}^i$ into a 2D matrix $\hat{\mathbf{C}}^i$ (row-wise), and concatenating the

result with the column vector $\bar{\mathbf{f}}^i$ whose elements are $\bar{f}^i(c)$, $c \in \mathcal{C}$, that we summarize as

$$\mathcal{T}^i = \left(\hat{\mathbf{C}}^i \,\middle\|\, \bar{\mathbf{f}}^i\right). \tag{3.8}$$

To be precise, any $\mathcal{T}^i$ is more like an empirical truth table than a classic one corresponding to an n-ary boolean function, indeed $\mathcal{T}^i$ can have repeated rows and missing Boolean tuple entries. However, $\mathcal{T}^i$ can be used to generate logic explanations in the same way, as we will explain in the next paragraph.

3.3.3 Extraction of logic explanations

Each row of the truth table $\mathcal{T}^i$ can be partitioned into two parts that are a binary tuple of concept activations, $\hat{c} \in \hat{\mathcal{C}}^i$, and the outcome of $\hat{f}^i(\hat{c}) \in \{0, 1\}$. An *example-level* logic formula, consisting in a single minterm, can be trivially extracted from each row for which $\hat{f}^i(\hat{c}) = 1$, by simply connecting with the logic AND $\wedge$ the true concepts and negated instances of the false ones. The logic formula becomes human understandable whenever concepts appearing in such a formula are replaced with human-interpretable strings that represent their name (similar consideration holds for $\hat{f}^i$, in what follows). For example, the following logic formula φ^i_q,

$$\varphi^i_q = \mathbf{c}_1 \wedge \ \neg\mathbf{c}_2 \wedge \ldots \wedge \mathbf{c}_{m_i}, \tag{3.9}$$

is the formula extracted from the q-th row of the table where, in the considered example, only the second concept is false, being $\mathbf{c}_2$ the name of the 2-nd concept. Example-level formulas can be aggregated with the logic OR $\vee$ to provide a *class-level* formula,

$$\bigvee_{t \in S_i} \varphi^i_t, \tag{3.10}$$

being $\mathcal{S}_i$ the set of rows indices of the truth table for which $\hat{f}^i(\hat{c}) = 1$, i.e. it is the support of $\hat{f}^i$. We define with $\phi^i(\hat{c})$ the function that holds true whenever Eq. 3.10, evaluated on a given Boolean tuple $\hat{c}$, is true. Due to the aforementioned definition of support, we get the following class-level first-order logic (FOL) explanation for all the concept tuples,

$$\forall \hat{c} \in \hat{\mathcal{C}}^i : \ \phi^i(\hat{c}) \leftrightarrow \hat{f}^i(\hat{c}). \tag{3.11}$$

We note that in case of non-concept-like input features, we may still derive the FOL formula through the "concept encoder" function g,

$$\forall x \in X : \ \phi^i\left(\xi(\overline{g(x)}, \mu^i)\right) \leftrightarrow \hat{f}^i\left(\xi(\overline{g(x)}, \mu^i)\right) \tag{3.12}$$

An example of the above scheme for both example and class-level explanations is depicted on top-right of Fig. 3.3.

Remark 3.3.1. The aggregation of many example-level explanations may increase the length and the complexity of the FOL formula being extracted for a whole class. However, existing techniques as the Quine–McCluskey algorithm can be used to get compact and simplified equivalent FOL expressions (McColl, 1878; Quine, 1952; McCluskey, 1956). For instance, the explanation $(person \wedge nose) \vee (\neg person \wedge nose)$ can be formally simplified in *nose*. Moreover, the Boolean interpretation of concept tuples may generate colliding representations for different samples. For instance, the Boolean representation of the two samples $\{(0.1, 0.7), (0.2, 0.9)\}$ is the tuple $\bar{c} = (0, 1)$ for both of them. This means that their example-level explanations match as well. However, a concept can be eventually split into multiple finer grain concepts to avoid collisions. Finally, we mention that the number of samples for which any example-level formula holds (i.e. the support of the formula) is used as a measure of the explanation importance. In practice, example-level formulas are ranked by support and iteratively aggregated to extract class-level explanations, until the aggregation improves the support of the formula on a validation set.

3.3.4 Loss function

The entropy of the probability distribution α^i (Eq. 3.4),

$$H(\alpha^i) = -\sum_{j=1}^{k} \alpha_j^i \log \alpha_j^i \tag{3.13}$$

is minimized when a single α_j^i is one, thus representing the extreme case in which only one concept matters, while it is maximum when all concepts are equally important. When H is jointly minimized with the usual loss function for supervised learning $\mathcal{L}_{\text{CrossEntr}}(y, f(c))$ (being y the target labels–we used the cross-entropy in our experiments), it allows the model to find a trade off between fitting quality and a parsimonious activation of the concepts, allowing each network f^i to predict i-th class memberships using few relevant concepts only. Overall, the loss function to train the network f is defined as,

$$\mathcal{L} \triangleq \mathbb{E}_{(c,y)} \left[\mathcal{L}_{\text{CrossEntr}}\Big(y, f(c)\Big) + \beta \sum_{i=1}^{r} H(\alpha^i) \right] \tag{3.14}$$

where $\beta > 0$ is the hyperparameter used to balance the relative importance of low-entropy solutions in the loss function. Higher values of β lead to sparser configuration of α, constraining the network to focus on a smaller set of concepts for each classification task (and vice versa), thus encoding the inductive human bias towards simple explanations (Miller, 1956; Cowan, 2001; Ma et al., 2014). It may be pointed out that a similar

regularization effect could be achieved by simply minimizing the L_1 norm over γ^i. However, the L_1 loss does not sufficiently penalize the concept scores for those features which are uncorrelated with the predicted category. The entropy loss, instead, correctly shrink to zero concept scores associated to uncorrelated features while the other remains close to one.

3.4 Evaluating logic explanations

While measuring classification metrics is necessary for models to be useful in solving classification tasks, assessing the quality of the explanations is required to justify their use for explainability. In contrast with other kinds of explanations, logic-based formulae can be evaluated quantitatively. To this end, we first train a LEN model to solve a given task. The training procedure designed for LENs allows the extraction of meaningful logic rules that serve as explanations as well as "rule-based classifiers". Hence, after the training stage we can evaluate the quality of the extracted rules on test data in terms of several well-established and sound metrics:

- *Explanation accuracy*: it measures how well the extracted logic formula φ_i correctly identifies the target class $Acc(\varphi_i, \bar{y}_i)$.

- *Complexity of an explanation*: it measures how hard would it be for a human being to understand the logic formula φ_i. This is simulated by standardizing the explanations in disjunctive normal form $\tilde{\varphi}$ and then by counting the average number of terms of the standardized formula $|\{\ell : \ \ell \text{ is a literal of } \varphi_i\}|$.

- *Fidelity of an explanation*: it measures how well the predictions obtained by applying the extracted explanations φ_i match the predictions obtained by using the classifier. When the LEN f_i is the classifier itself (i.e., interpretable classification), this metric represents the match between the extracted explanation and the LEN prediction $Acc(\varphi_i, \bar{f}_i)$. Instead, when the LEN is explaining the predictions of a black-box classifier g_i, this metric represents the agreement between the extracted explanation and the prediction of black-box classifier $Acc(\varphi_i, \bar{g}_i)$.

- *Consistency of an explanation*: it measures the similarity of the extracted explanations over different runs. It is computed by counting how many times the same concepts appear in the logic formulas over different folds of a K-fold cross-validation.

3.5 Experiments

3.5.1 Research questions

In this section, we analyze the performance of logic explained networks by means of the following research questions:

- **Task generalization** — How does the entropy LEN generalize on unseen samples compared to existing rule-based white-box models?
- **Explainability** — Are logic explanations accurate when used to classify unseen samples? Are logic explanations simple? Are logic explanations shorter or longer w.r.t. logic sentences generated by existing rule learners? Are logic rules faithful w.r.t. the neural model i.e., do their predictions match the predictions of the neural model?
- **Efficiency and robustness** — How long does it take for LENs to generate logic rules w.r.t. existing rule learners? Do LENs generate similar logic rules under different initialization conditions and data splits?

3.5.2 Datasets

We investigate our research questions using four classification problems ranging from computer vision to medicine. In particular we use: (i) The Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II, (Saeed et al., 2011; Goldberger et al., 2000)), an intensive care unit database where the task is to predict whether subjects will recover after the hospitalization using a variety of clinical features (e.g., physiological, biochemical); (ii) the Varieties of Democracy (V-Dem, (Pemstein et al., 2018; Coppedge et al., 2021)), a dataset where the task is to identify democratic countries based on a collection of indicators of latent regime characteristics (e.g. freedom of expression, freedom of association, equality before the law, etc) over 202 countries from 1789 to 2020. (iii) The Modified National Institute of Standards and Technology database (MNIST, (LeCun, 1998)), a large collection of images representing handwritten digits. In our experiments we designed a new task which is to determine whether an image contains an even or an odd digit and where concepts are digit labels (e.g., 0, 1, 2, ...). (iv) The Caltech-UCSD Birds-200-2011 dataset (CUB, (Wah et al., 2011)) is a fine-grained classification dataset including 112 bird attributes describing visual characteristics (color, pattern, shape) of particular parts (beak, wings, tail, etc.) and 200 task labels corresponding to specific bird species. For computer vision datasets (e.g. CUB) we use ResNet15 as concept encoders to map raw images into a concept space, following Koh et al. (2020). In the other datasets, we scale the input data

into a categorical space suitable for concept-based models. All datasets can be downloaded from publicly available resources.

3.5.3 Baselines

We compare entropy-based LENs against state-of-the-art white-box models providing global logic explanations in order to compare all models using the same metrics. For this reason we do not include in our comparison white-box models providing local explanations (Guidotti et al., 2018; Ribeiro et al., 2018) or feature rankings such as logistic regression or Generalized Additive Models (Hastie, 2017). We focus instead on comparing LENs with rule-based systems such as Decision Trees (DT, (Breiman et al., 1984)) and powerful rule-mining approaches such as Bayesian Rule Lists (BRL, (Letham et al., 2015)), where a set of rules is "pre-mined" using the frequent-pattern tree mining algorithm (Han et al., 2000) and then the best rule set is identified with Bayesian statistics. In our experiments we also include the ψ net (Ciravegna et al., 2020), a previously published neural architecture providing logic explanations for its predictions. As a baseline reference we include in our comparison a black-box neural network with the same learning capacity (number of layers and parameters) w.r.t. the LEN models.

3.5.4 Metrics

We measure model's performance based on six metrics. First we measure a model classification performance in terms of its *task accuracy*. Having extracted logic explanations we then measure the quality of the explanations by using them as "rule classifiers" by computing the *rule task accuracy*. We also compute the *complexity* of the logic rules to estimate how hard it would be for a human to understand the logic expression. To evaluate rule robustness and faithfulness with respect to the neural model, we compute the rule *consistency* and *fidelity*, respectively. Finally, we assess the *time* required to extract the full rule set for each model. All metrics in our evaluation, across all experiments, are computed on test sets using 5 random seeds, from which we compute a metric's mean and 95% confidence interval.

3.6 Results and discussion

3.6.1 Task generalization

LENs attain competitive task accuracy w.r.t. state-of-the-art rule learners (Table 3.1) Our results show that entropy-based LENs outperform state-of-the-art white-box models such as BRL and decision trees and interpretable neural models such as

ψ networks on challenging datasets. The reason why the proposed approach consistently outperform ψ networks can be explained observing how entropy-based networks are far less constrained than ψ networks, both in the architecture and in the loss function. Indeed, ψ nets apply a strong weight regularization in all hidden layers and ultimately prunes weights with low values during training. As a result, the resulting learning capacity of the neural architecture is significantly diminished with a detrimental impact on task accuracy. Similarly, the main reason why entropy-based LENs provide a higher classification accuracy with respect to BRL and decision trees may lie in the smoothness of the decision functions of neural networks which tend to generalize better than rule-based methods, as already observed by Tavares et al. (2020). Finally, entropy-based LENs attain similar task accuracy when compared to an equivalent black-box neural network even on challenging benchmarks such as V-Dem or CUB. This suggests that the entropy layer and the entropy-based regularization have only minor effects on the learning capacity of the model, while they provide a significant improvement in terms of explainability.

Table 3.1: Classification accuracy (%) of the compared models.

	Entropy net	Tree	BRL	ψ net	Black box
MIMIC-II	$\mathbf{79.05 \pm 1.35}$	77.53 ± 1.45	76.40 ± 1.22	77.19 ± 1.64	77.81 ± 2.45
V-Dem	94.51 ± 0.48	85.61 ± 0.57	91.23 ± 0.75	89.77 ± 2.07	$\mathbf{94.53 \pm 1.17}$
MNIST	$\mathbf{99.81 \pm 0.02}$	99.75 ± 0.01	99.80 ± 0.02	99.79 ± 0.03	99.81 ± 0.08
CUB	92.95 ± 0.20	81.62 ± 1.17	90.79 ± 0.34	91.92 ± 0.27	$\mathbf{93.32 \pm 0.35}$

3.6.2 Explainability

LENs generate simple and accurate logic explanations (Figure 3.4) Optimal logic explanations must be simultaneously simple and accurate. For this reason we mainly evaluate the quality of the explanations in a combined view of two of these two main metrics, reporting the Pareto frontiers (Marler and Arora, 2004) for each experiment in terms of the explanation and model error (100 minus the explanation/model accuracy, see Figure 3.4). From this graphical representation we can easily realize that logic explanations generated by entropy-based LENs represent non-dominated solutions (Marler and Arora, 2004) compared to BRL and decision trees. Indeed, the logic formulae extracted from entropy-based networks are either better or almost as accurate as the formulae found by decision trees or mined by BRL (even if BRL attains the highest performance). However, the complexity of the rules generated by LENs is significantly lower than BRL. Notice how less complex rules implies more readable formulas, that is a crucial feature in the context of explainable AI. As a result, our experiments suggest that LENs formulae might be preferable with respect to BRL whenever a simple explanation is required to understand

the key elements that play a significant role in solving the given task. More specifically, the complexity of LENs explanations is usually lower than the complexity of the rules extracted both from a decision tree[1] or mined by BRL. This is mostly due to the use of the entropy layer and entropy regularization which contribute in keeping low the explanation complexity of LENs. Indeed, by selecting a limited number of concepts and by filtering out low-frequency example-level explanations, LENs can actually produce readable rules whose size is bounded, while attaining competitive classification performances. Comparing entropy-based LENs with ψ networks, we observe that ψ networks yield moderately complex explanations and often a higher classification error with respect to entropy-based networks. Again, this confirms how the neural architecture and regularization embedded in entropy-based LENs is far less restrictive compared to ψ networks, allowing the model to achieve better performances.

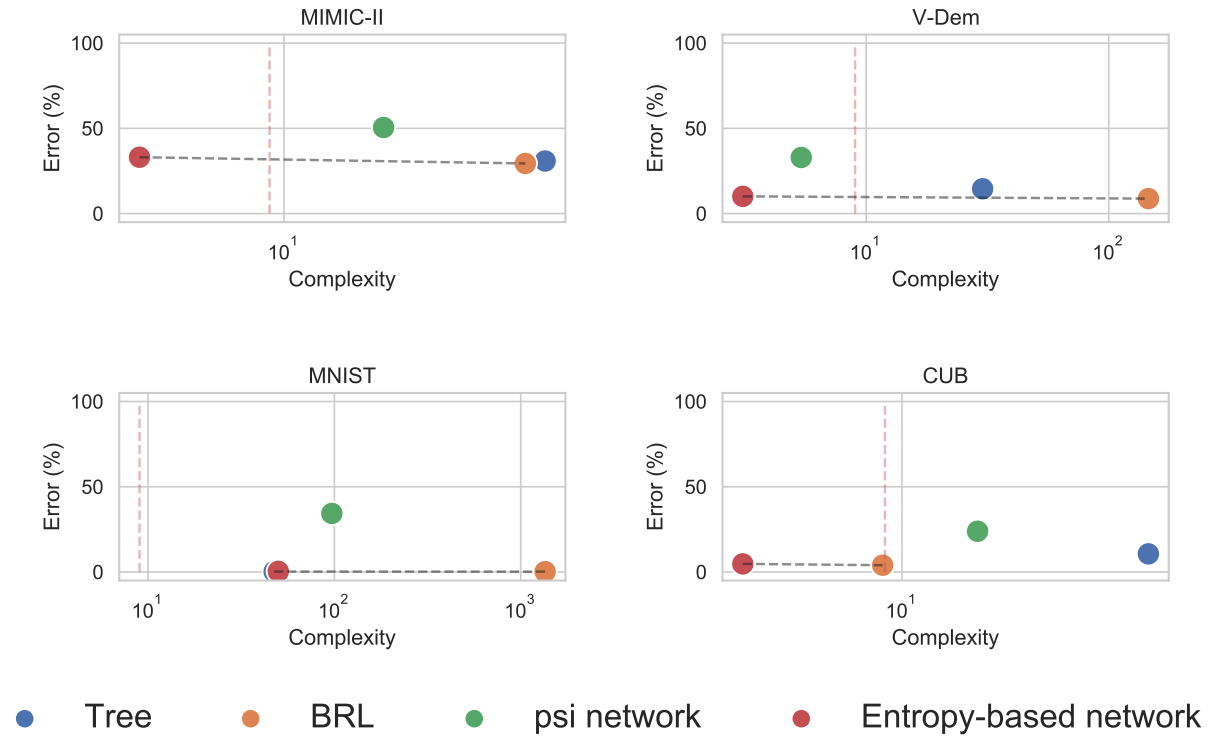


Figure 3.4: Non-dominated solutions (Marler and Arora, 2004) (dotted black line) in terms of average explanation complexity and average explanation test error. The vertical dotted red line marks the maximum explanation complexity laypeople can handle (i.e. complexity $\approx$ 9, see (Miller, 1956; Cowan, 2001; Ma et al., 2014)). Notice how the explanations provided by the Entropy-based Network are always one of the non-dominated solution.

LENs generate faithful explanations (Table 3.2) Moving to more fine-grained details, our experiments show how the fidelity of the formulae extracted by the entropy-

[1]Decision trees have been limited to a maximum of 5 decision levels in order to extract rules of comparable length with the other methods.

based network is always higher than 90% with the only exception of MIMIC. This means that almost any prediction made using the logic explanation matches the corresponding prediction made by the model, making the proposed approach very close to a white box model. However white-box models, like decision trees and BRL, trivially outperform LENs in terms of fidelity. This is due to the fact that such models make predictions directly based on explanations. Therefore fidelity is (trivially) always 100%. On the contrary, the fidelity of the formulas extracted by the ψ network is usually significantly lower. This is mostly due to the fact that ψ networks extract a formula for every neuron of the net, and then generate compound explanations composing logic rules layer-by-layer up to the output neurons. As a result, even though the behaviour of each neuron of the ψ network can be interpreted, the overall logic formula may be a less precise explanation of the model predictions, since each neuron-related explanation might introduce errors that propagate when composing them. This enables ψ networks to have a more fine-grained level of explanations, while hampering the quality of global logic rules. Finally, a notable mention on explanation fidelity comes from the MNIST experiment. Indeed entropy-based LENs represent the only model able to match the ground-truth explanation for this experiment, i.e. $\forall x, \text{isOdd(x)} \leftrightarrow \text{isOne(x)} \oplus \text{isThree(x)} \oplus \text{isFive(x)} \oplus \text{isSeven(x)} \oplus \text{isNine(x)}$ and $\forall x, \text{isEven(x)} \leftrightarrow \text{isZero(x)} \oplus \text{isTwo(x)} \oplus \text{isfour(x)} \oplus \text{isSix(x)} \oplus \text{isEight(x)}$, being $\oplus$ the exclusive OR.

	Entropy net	ψ net
MIMIC-II	$\mathbf{79.11 \pm 2.02}$	51.63 ± 6.67
V-Dem	$\mathbf{90.90 \pm 1.23}$	69.67 ± 10.43
MNIST	$\mathbf{99.63 \pm 0.00}$	65.68 ± 5.05
CUB	$\mathbf{99.86 \pm 0.01}$	77.34 ± 0.52

Table 3.2: Explanation fidelity (%).

3.6.3 Efficiency and robustness

LENs generate consistent explanations (Table 3.3) In terms of consistency, our experiments suggest that LENs provide logic explanations which are slightly more stable across different parameter initialization and data splits compared to ψ networks. However, overall BRL outperforms all the other methods in terms of rule consistency—closely followed by entropy-based LENs—and that rule consistency is significantly impacted by the given dataset/task. Our intuition is that those datasets that are more coherently represented by the data in the different folds are expected to lead to more consistent behaviors. Most likely, BRL is able to cope slightly better with these data distribution shifts as it learning procedure optimizes rule stability.

	Entropy net	Tree	BRL	ψ net
MIMIC-II	28.75	**40.49**	30.48	27.62
V-Dem	46.25	72.00	**73.33**	38.00
MNIST	**100.00**	41.67	**100.00**	96.00
CUB	35.52	21.47	**42.86**	41.43

Table 3.3: Explanation consistency (%).

LENs rapidly extract logic explanations (Table 3.5) The time required to extract logic explanations from entropy-based networks is only slightly higher with respect to Decision Trees but it is lower than ψ Networks and BRL by one to three orders of magnitude. This makes entropy-based LENs more suitable for explaining complex tasks and mine rule efficiently on larger data sets. Overall, BRL is the slowest rule extractor across all the experiments. In two cases, BRL takes about 1 hour to extract an explanation and over 1 day in the case of CUB, making it unsuitable for supporting a fast extraction of explanations. Decision trees are the fastest model overall as their learning process is quite optimized for extracting simple rules and, once extracted, rules can be directly applied on test data. On the contrary, entropy-based networks they first need to be trained and then the trained network needs to be analyzed to extract logic explanations, thus introducing a computational overhead which slows the generation of explanations down. However, this computational overhead has a minor impact as it does not change the order of magnitude of the time required to extract logic rules.

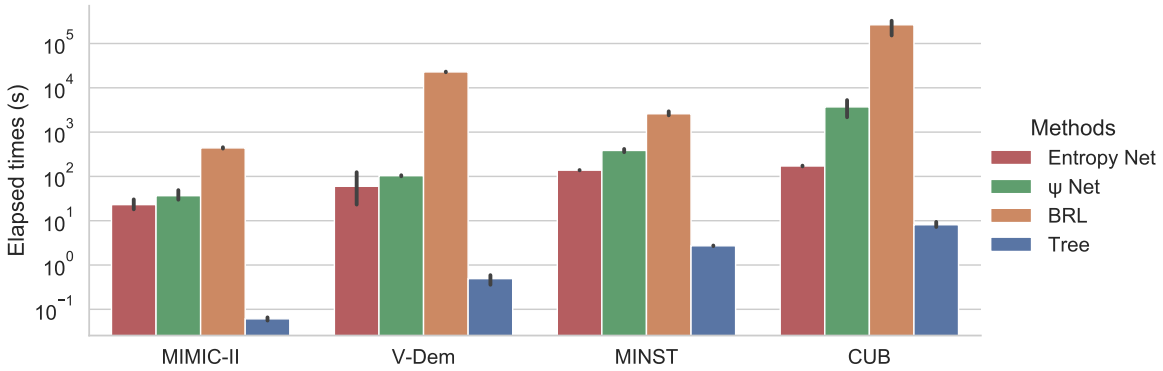


Figure 3.5: Time required to train models and to extract the explanations. Our model compares favorably with the competitors, with the exception of Decision Trees. BRL is by one to three order of magnitude slower than our approach. Error bars show the 95% confidence interval of the mean.

3.7 Key findings and limitations

Overall, the results of our experiments demonstrate how LENs can address the **Aim #1** of this work (see Section 2.5) as they:

- attain better or competitive task accuracy w.r.t. existing concept-based models;
- generate accurate compound logic explanations which tend to be much more concise than those provided by existing white-box rule learners (e.g., decision trees and Bayesian rule lists)

In fact, LENs represents a scalable, self-explaining neural approach providing first-order logic explanations for its predictions. The sparse attention mechanism allows these models to select only the most relevant concepts thus providing simple and accurate logic explanations. This way, users of concept-based models can verify whether the model is learning as intended by checking which concepts are selected and how they are used to form a prediction.

However, while LENs provide a good compromise between accuracy and explainability, they still struggle to attain state-of-the-art performances in solving complex tasks outperforming black-boxes in terms of task accuracy which represents the **Aim #2** of this work. To address this, in the next chapter we discuss the main limitations preventing concept-based models to outperform black-boxes in terms of task accuracy. We then use the results of our analysis to address the **Aim #2** of this work by designing concept-based models attaining state-of-the-art task performance.

Chapter 4

Concept embeddings (beyond the accuracy-explainability trade-off)

Motivation—In the previous chapter we discussed how to design the architecture of neural networks to allow the extraction of simple logic explanations. While logic explained networks address the **Aim #1** of this work, they still struggle in attaining state-of-the-art task accuracy, outperforming equivalent black-box models. This condition is commonly known as the "accuracy-explainability trade-off". This kind of issue is one of the key concerns and main research topics in explainable AI as these models often struggle to provide a good compromise between the accuracy of their predictions and the quality of their explanations.

Solution—To fill this knowledge gap, in this chapter we will present Concept Embedding Models (CEMs, (Zarlenga et al., 2022)), a novel class of CBMs aiming to:

- break the accuracy-explainability trade-off in concept-based models;
- scale concept-based models to real-world conditions where concept supervisions are scarce and noisy;

The **key innovation** of CEMs is a fully supervised high-dimensional concept representation. This high-dimensional representation increases the capacity of CEMs at concept level. The increased model capacity allows to encode more information in each concept beyond the probability of a concept being active/inactive, including contextual nuances which CEMs can use to have a deeper understanding of each concept and to solve tasks more efficiently.

In this chapter we will discuss the main limitations of concept bottleneck models (Section 4.1). We will then formally describe state-of-the-art concept-based architectures (Section 4.2) and present our novel approach i.e., Concept Embedding Models (Section 4.3). The remaining of the chapter: (i) describes the key metrics to evaluate Concept Embedding Models (Section 4.4), (ii) illustrates the experimental setup we use to benchmark these

models (Section 4.5), (iii) discusses the results of the experiments (Section 4.6), and (iv) summarizes the key findings and limitations of the proposed approach (Section 4.7).

4.1 What is wrong with "concept bottlenecks"?

The accuracy-explainability trade-off represents the main limitation of all explainable AI models as they struggle between two objectives when solving complex tasks: provide simple and robust explanations and attain high task accuracy. Even state-of-the-art concept-based models, such as LENs, can achieve optimal solutions for two objectives for simple tasks. However, in real-world conditions they do not escape this doom.

Beyond LENs, this limitation affects a broader set of concept-based architectures commonly known as "Concept Bottleneck Models" (Koh et al., 2020) (CBM). These neural architectures are characterized by what we call a "concept bottleneck" which represents a human-interpretable interface between the concept encoder and the concept-based model generating task predictions. As discussed in the previous chapters, these concept interfaces enable neural architectures to provide simple explanations and allow human experts to improve model performance by correcting mispredicted concepts through test-time "concept interventions". Unfortunately, concept bottlenecks may impair task accuracy (Koh et al., 2020; Mahinpei et al., 2021), especially in real-world conditions e.g., when concept labels do not contain all the necessary information for accurately predicting a downstream task (i.e., they form an "incomplete" representation of the task (Yeh et al., 2020)), as seen in Figure 4.1.

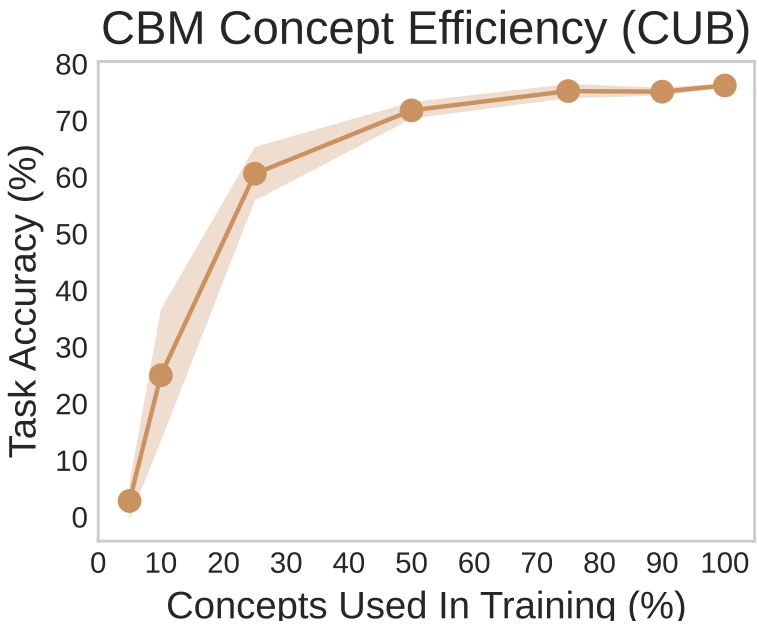


Figure 4.1: Real-world conditions (e.g., concept incompleteness and noisy supervisions) impair Concept Bottleneck Models (CBMs) task accuracy. As concept supervisions available for training are reduced, the task accuracy drops dramatically.

To overcome this limitation, Mahinpei et al. (2021) proposed to augment the learning capacity of CBMs by introducing extra neurons at concept level whithout imposing any direct concept supervision on their activations. This solution (a.k.a. "hybrid CBMs") allows the model to efficiently solve tasks even in noisy and concept-incomplete settings. However, this performance improvement comes with a cost: test-time concept interventions become ineffective. This suggests that hybrid CBMs are prone to the phenomenon known as "shortcut learning": task performance is independent from concept activations as it relies mostly on the unsupervised extra neurons. This result demonstrates that hybrid CBMs cannot provide reliable concept-based explanations for task predictions nor they can effectively interact with human experts through the learnt concepts.

4.2 Concept bottlenecks: data structures and interventions

Before diving into the details of concept embeddings, we first summarize the essential information required to understand the state-of-the-art of concept bottlenecks. To this end, in the next paragraphs we briefly describe the details of the main concept data structures/representations and concept intervention strategies allowing human experts to interact with and improve CBMs.

Concept representations Recall from Chapter 2 that for each sample $\mathbf{x} \in \mathcal{X}$, the concept encoder g learns k different scalar concept representations $\hat{c}_1, \ldots, \hat{c}_k$. Boolean and Fuzzy CBMs (Koh et al., 2020) are concept-based architectures assuming that each dimension of $\hat{\mathbf{c}}$, which we describe by $\hat{c}_i = s(\hat{\mathbf{c}})_{[i]} \in [0, 1]$, is aligned with a single ground truth concept and represents a probability of that concept being active. To model this probability, these architectures employ the element-wise activation function $s : \mathbb{R} \to [0, 1]$ on the concept values predicted by the concept encoder. This activation can be either a thresholding function $s(x) \triangleq \mathbb{1}_{x \geq 0.5}$, generating what we refer to as *Boolean* CBM, or sigmoidal function $s(x) \triangleq 1/(1 + e^{-x})$, generating what we refer to as *Fuzzy* CBM.[1] A natural extension of this framework is a *Hybrid* CBM (Mahinpei et al., 2021), where $\hat{\mathbf{c}} \in \mathbb{R}^{(k+\gamma)}$ contains γ unsupervised dimensions and k supervised concept dimensions which, when concatenated, form a shared concept vector (i.e., an "embedding").

Concept interventions Interventions are one of the core motivations behind CBMs (Koh et al., 2020). Through interventions, concept bottleneck models allow domain experts to improve a CBM's task performance by rectifying mispredicted concepts by setting, at test-time, $\hat{c}_i := c_i$ (where c_i is the ground truth value of the i-th concept). Such interventions can significantly improve CBMs performance within a human-in-the-loop setting (Koh et al., 2020). Furthermore, interventions enable the construction of meaningful concept-based counterfactuals (Wachter et al., 2017). For example, intervening on a CBM trained to predict bird types from images can determine that when the size of a "black" bird with "black" beak changes from "medium" to "large", while all other concepts remain constant, then one may classify the bird as a "raven" rather than a "crow".

The challenge we address in the next section is to design a concept representation such that CBMs can attain state-of-the-art task accuracy while preserving the ability to provide simple concept-based explanations and to allow effective interventions.

4.3 Concept embedding models

In real-world settings, where complete concept annotations are costly and rare, vanilla CBMs may need to compromise their task performance in order to preserve their interpretability (Koh et al., 2020). While Hybrid CBMs are able to overcome this issue by adding extra capacity in their bottlenecks, this comes at the cost of their interpretability and their responsiveness to concept interventions, thus undermining user trust (Shen, 2022). To overcome these pitfalls, we propose *Concept Embedding Models* (CEMs), a concept-based architecture which represents each concept as a supervised vector. Intu-

[1] In practice (e.g., (Koh et al., 2020)) one may use logits rather than sigmoidal activations to improve gradient flow (Glorot et al., 2011). However, the structure of the bottleneck does not change.

itively, using high-dimensional embeddings to represent each concept allows for extra *supervised* learning capacity, as opposed to Hybrid models where the information flowing through their *unsupervised* bottleneck activations is concept-agnostic. In the following section, we introduce our architecture and describe how it learns a mixture of two semantic embeddings for each concept (Figure 4.2). We then discuss how interventions are performed in CEMs and introduce *RandInt*, a train-time regularisation mechanism that incentivises our model to positively react to interventions at test-time.

4.3.1 Mixture of concept embeddings

For each concept, CEM learns a mixture of two embeddings with explicit semantics representing the concept's activity. Such design allows our model to construct evidence both in favour of and against a concept being active, and supports simple concept interventions as one can switch between the two embedding states at intervention time.

More specifically, we represent concept c_i with two embeddings $\hat{\mathbf{c}}_i^+, \hat{\mathbf{c}}_i^- \in \mathbb{R}^m$, each with a specific semantics: $\hat{\mathbf{c}}_i^+$ represents its active state (concept is `true`) while $\hat{\mathbf{c}}_i^-$ represents its inactive state (concept is `false`). To this aim, a deep neural network $\psi(\mathbf{x})$ learns a latent representation $\mathbf{h} \in \mathbb{R}^{n_{\text{hidden}}}$ which is the input to any CEM. MixCEM then feeds $\mathbf{h}$ into two concept-specific fully connected layers, which learn two concept embeddings in $\mathbb{R}^m$, namely $\hat{\mathbf{c}}_i^+ = \phi_i^+(\mathbf{h}) = a(W_i^+\mathbf{h} + \mathbf{b}_i^+)$ and $\hat{\mathbf{c}}_i^- = \phi_i^-(\mathbf{h}) = a(W_i^-\mathbf{h} + \mathbf{b}_i^-)$.[2] Notice that while more complicated architectures/models can be used to parameterise our concept embedding generators $\phi_i^+(\mathbf{h})$ and $\phi_i^-(\mathbf{h})$, we opted for a simple one-layer neural network to constrain parameter growth in models with large bottlenecks.

[2]In practice, we use a leaky-ReLU for the activation $a(\cdot)$

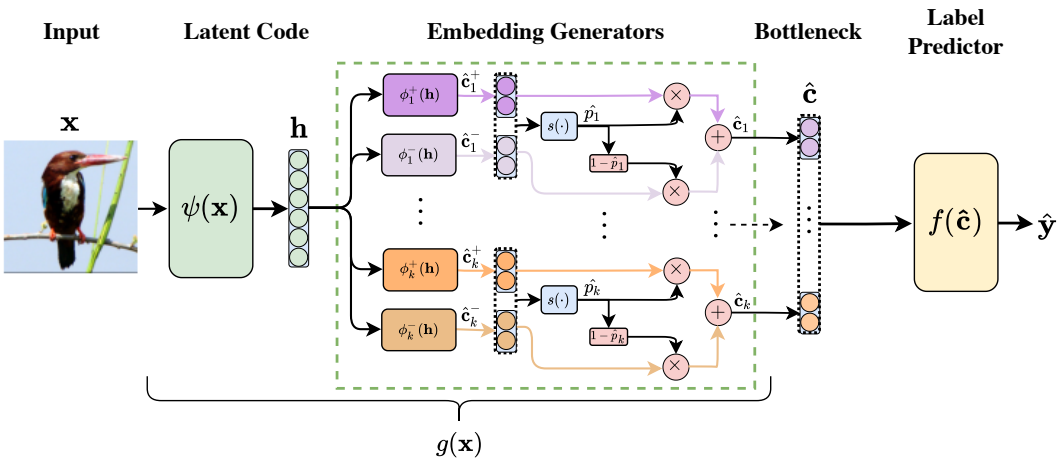


Figure 4.2: **Concept Embeddings Model**: from an intermediate latent code $\mathbf{h}$, we learn two embeddings per concept, one for when it is active (i.e., $\hat{\mathbf{c}}_i^+$), and another when it is inactive (i.e., $\hat{\mathbf{c}}_i^-$). Each concept embedding (shown in this example as a vector with $m = 2$ activations) is then aligned to its corresponding ground truth concept through the scoring function $s(\cdot)$, which learns to assign activation probabilities $\hat{p}_i$ for each concept. These probabilities are used to output an embedding for each concept via a weighted mixture of each concept's positive and negative embedding.

Our architecture encourages embeddings $\hat{\mathbf{c}}_i^+$ and $\hat{\mathbf{c}}_i^-$ to be aligned with ground-truth concept c_i via a learnable and differentiable scoring function $s : \mathbb{R}^{2m} \rightarrow [0, 1]$, trained to predict the probability $\hat{p}_i$ of concept c_i being active from the embeddings' joint space, i.e., $\hat{p}_i \triangleq s([\hat{\mathbf{c}}_i^+, \hat{\mathbf{c}}_i^-]^T) = \sigma\big(W_s[\hat{\mathbf{c}}_i^+, \hat{\mathbf{c}}_i^-]^T + \mathbf{b}_s\big)$. We constrain parameters W_s and $\mathbf{b}_s$ to be shared across all concepts for parameter efficiency. We construct the final concept embedding $\hat{\mathbf{c}}_i$ for c_i as a weighted mixture of $\hat{\mathbf{c}}_i^+$ and $\hat{\mathbf{c}}_i^-$ as:

$$\hat{\mathbf{c}}_i \triangleq \big(\hat{p}_i \hat{\mathbf{c}}_i^+ + (1 - \hat{p}_i)\hat{\mathbf{c}}_i^-\big)$$

Intuitively, this serves a two-fold purpose: (i) it forces the model to depend only on $\hat{\mathbf{c}}_i^+$ when the i-th concept is active, i.e., $c_i = 1$ (and only on $\hat{\mathbf{c}}_i^-$ when inactive), leading to two different semantically meaningful latent spaces, and (ii) it enables a clear intervention strategy where one switches the embedding states when correcting a mispredicted concept, as discussed below.

Finally, MixCEM concatenates all k mixed concept embeddings, resulting in a bottleneck $g(\mathbf{x}) = \hat{\mathbf{c}}$ with $k \cdot m$ units (see end of Figure 4.2). This is passed to the label predictor f to obtain a downstream task label.

4.3.2 Intervening with Concept Embeddings

As in vanilla CBMs, MixCEMs support test-time concept interventions. To intervene on concept c_i, one can update $\hat{\mathbf{c}}_i$ by swapping the output concept embedding for the

one semantically aligned with the concept ground truth label. For instance, if for some sample $\mathbf{x}$ and concept c_i a MixCEM predicted $\hat{p}_i = 0.1$ while a human expert knows that concept c_i is active ($c_i = 1$), they can perform the intervention $\hat{p}_i := 1$. This operation updates MixCEM's bottleneck by setting $\hat{\mathbf{c}}_i$ to $\hat{\mathbf{c}}_i^+$ rather than $\left(0.1\hat{\mathbf{c}}_i^+ + 0.9\hat{\mathbf{c}}_i^-\right)$. Such an update allows the downstream label predictor to act on information related to the corrected concept.

In addition, we introduce *RandInt*, a regularisation strategy exposing MixCEMs to concept interventions during training to improve the effectiveness of such actions at test-time. RandInt randomly performs independent concept interventions during training with probability p_{int} (i.e., $\hat{p}_i$ is set to $\hat{p}_i := c_i$ for concept c_i with probability p_{int}). In other words, for all concepts c_i, their embeddings during training are computed as:

$$\hat{\mathbf{c}}_i = \begin{cases} \left(c_i\hat{\mathbf{c}}_i^+ + (1 - c_i)\hat{\mathbf{c}}_i^-\right) & \text{with probability } p_{\text{int}} \\ \left(\hat{p}_i\hat{\mathbf{c}}_i^+ + (1 - \hat{p}_i)\hat{\mathbf{c}}_i^-\right) & \text{with probability } (1 - p_{\text{int}}) \end{cases}$$

while at test-time we always use the predicted probabilities for performing the mixing. During backpropagation, this strategy forces feedback from the downstream task to update only the correct concept embedding (e.g., $\hat{\mathbf{c}}_i^+$ if $c_i = 1$) while feedback from concept predictions can update both $\hat{\mathbf{c}}_i^+$ and $\hat{\mathbf{c}}_i^-$. Under this view, RandInt can be thought of as learning an average over an exponentially large family of MixCEM models (similarly to dropout (Srivastava et al., 2014)) where some of the concept representations are trained using only feedback from their concept label while others receive training feedback from both their concept and task labels. In the extreme case when the embedding size is $m = 1$ and we only have one concept (i.e., $k = 1$), this process can be seen as randomly alternating between learning a Joint-CBM and a Sequential-CBM during training, with p_{int} controlling how often we switch between joint training and sequential training.

4.3.3 Loss function

In practice, following Koh et al. (2020), we use an interpretable label predictor f parameterised by a simple linear layer, though more complex functions could be explored too. Notice that as in vanilla CBMs, MixCEM provides a concept-based explanation for the output of f through its concept probability vector $\hat{\mathbf{p}} \triangleq [\hat{p}_1, \cdots, \hat{p}_k]$, indicating the predicted concept activity. This architecture can be trained in an end-to-end fashion by jointly minimizing via stochastic gradient descent a weighted sum of the cross entropy loss on both task prediction and concept predictions:

$$\mathcal{L} \triangleq \mathbb{E}_{(\mathbf{x},y,\mathbf{c})}\left[\mathcal{L}_{\text{task}}\Big(y, f\big(g(\mathbf{x})\big)\Big) + \beta\mathcal{L}_{\text{CrossEntr}}\Big(\mathbf{c}, \hat{\mathbf{p}}(\mathbf{x})\Big)\right] \tag{4.1}$$

where hyperparameter $\beta \in \mathbb{R}^+$ controls how much we value concept accuracy w.r.t. downstream task accuracy.

4.4 Evaluating concept embeddings

To the best of our knowledge, while a great deal of attention has been paid to concept-based explainability in recent years, existing work still fails to provide methods that can be used to evaluate the interpretability of a concept embedding or to explain why certain CBMs underperform in their task predictions. With this in mind, we propose (i) a new metric for evaluating concept quality in multidimensional representations and (ii) an information-theoretic method which, by analysing the information flow in concept bottlenecks, can help understand why a CBM may underperform in a downstream task.

Concept Alignment Score (CAS) While concept predictive accuracy is well defined for scalar concept representations (e.g., vanilla CBMs), there seems to be no clear metric for evaluating the "concept accuracy" of an embedding representation. Therefore, in this work we build upon this gap and propose the Concept Alignment Score (CAS) as a generalization of the concept predictive accuracy. Intuitively, if a concept representation is able to capture a concept correctly, then we would expect that clustering samples based on that representation would result in coherent clusters where samples within the same cluster all have the concept active or inactive. The CAS attempts to capture this by looking at how coherent clusters are for each concept representation using the known concept labels for each sample as we change the size of each cluster. This is formally computed via Equation 4.4 throught a repeated evaluation of Rosenberg et al.'s homogeneity score (Rosenberg and Hirschberg, 2007) for different clusterings.

Following Rosenberg and Hirschberg (2007), we compute the homogeneity score as described in Section 4.4 by estimating the conditional entropy of ground truth concept labels $\mathcal{C}_i$ w.r.t. cluster labels Π_i, i.e. $H(\mathcal{C}_i, \Pi_i)$, using a contingency table. This table is produced by our selected clustering algorithm κ, i.e. $A = \{a_{u,v}\}$ where $a_{u,v}$ is the number of data points that are members of class $c_i = v \in \{0, 1\}$ and elements of cluster $\pi_i = u \in \{1, \cdots, \rho\}$:

$$H(\mathcal{C}_i, \Pi_i) = -\frac{\rho}{N} \sum_{u=1}^{\rho} \left(a_{u,0} \log \frac{a_{u,0}}{a_{u,0} + a_{u,1}} + a_{u,1} \log \frac{a_{u,1}}{a_{u,0} + a_{u,1}} \right) \tag{4.2}$$

Similarly, we compute the entropy of the ground truth concept labels $\mathcal{C}_i$, i.e. $H(\mathcal{C})$, as:

$$H(\mathcal{C}_i) = -\left(\frac{\sum_{u=1}^{\rho} a_{u,0}}{2} \log \frac{\sum_{u=1}^{\rho} a_{u,0}}{2} + \frac{\sum_{u=1}^{\rho} a_{u,1}}{2} \log \frac{\sum_{u=1}^{\rho} a_{u,1}}{2} \right) \tag{4.3}$$

When evaluating the CAS, we use $\delta = 50$ to speed up its computation across all datasets. The Concept Alignment Score (CAS) aims to measure how much learnt concept representations can be trusted as faithful representations of their ground truth concept labels. Intuitively, CAS generalises concept accuracy by considering the homogeneity of predicted concept labels within groups of samples which are close in a concept subspace. More specifically, for each concept c_i the CAS applies a clustering algorithm κ to find $\rho > 2$ clusters, assigning to each sample $\mathbf{x}^{(j)}$ a cluster label $\pi_i^{(j)} \in \{1, \cdots, \rho\}$. We compute this label by clustering samples using their i-th concept representations $\{\hat{\mathbf{c}}_i^{(1)}, \hat{\mathbf{c}}_i^{(2)}, \cdots\}$. Given N test samples, the homogeneity score $h(\cdot)$ (Rosenberg and Hirschberg, 2007) then computes the conditional entropy H of ground truth labels $\mathcal{C}_i = \{c_i^{(j)}\}_{j=1}^N$ w.r.t. cluster labels $\Pi_i(\kappa, \rho) = \{\pi_i^{(j)}\}_{j=1}^N$, that is, $h = 1$ when $H(\mathcal{C}_i, \Pi_i) = 0$ and $h = 1 - H(\mathcal{C}_i, \Pi_i)/H(\mathcal{C}_i)$ otherwise. The higher the homogeneity, the more a learnt concept representation is "aligned" with its labels, and can thus be trusted as a faithful representation. CAS averages homogeneity scores over all concepts and number of clusters ρ, providing a normalised score in $[0, 1]$:

$$\text{CAS}(\hat{\mathbf{c}}_1, \cdots, \hat{\mathbf{c}}_k) \triangleq \frac{1}{N-2} \sum_{\rho=2}^{N} \left(\frac{1}{k} \sum_{i=1}^{k} h(\mathcal{C}_i, \Pi_i(\kappa, \rho)) \right) \tag{4.4}$$

To tractably compute CAS in practice, we sum homogeneity scores by varying ρ across $\rho \in \{2, 2+\delta, 2+2\delta, \cdots, N\}$ for some $\delta > 1$. Furthermore, we use k-Medoids (Kaufman and Rousseeuw, 1990) for cluster discovery, as used in Ghorbani et al. (2019a) and Magister et al. (2021), and use concept logits when computing the CAS for Boolean and Fuzzy CBMs. For Hybrid CBMs, we use $\hat{\mathbf{c}}_i \triangleq [\hat{\mathbf{c}}_{[k:k+\gamma]}, \hat{\mathbf{c}}_{[i:(i+1)]}]^T$ as the concept representation for c_i given that the extra capacity is a shared embedding across all concepts.

Information bottleneck The relationship between the quality of concept representations w.r.t. the input distribution remains widely unexplored. Here we propose to analyse this relationship using information theory methods for DNNs developed by Tishby et al. (2000) and Tishby and Zaslavsky (2015). In particular, we compare concept bottlenecks using the Information Plane method (Tishby et al., 2000) to study the information flow at concept level. To this end, we measure the evolution of the Mutual Information ($I(\cdot, \cdot)$) of concept representations w.r.t. the input and output distributions across training epochs. We conjecture that embedding-based CBMs circumvent the information bottleneck by preserving more information than vanilla CBMs from the input distribution as part of their high-dimensional activations. If true, such effect should be captured by Information Planes in the form of a positively correlated evolution of $I(\mathcal{X}, \hat{\mathcal{C}})$, the Mutual Information (MI) between inputs $\mathcal{X}$ and learnt concept representations $\hat{\mathcal{C}}$, and $I(\hat{\mathcal{C}}, \mathcal{Y})$, the MI between learnt concept representations $\hat{\mathcal{C}}$ and task labels $\mathcal{C}$. In contrast, we anticipate that scalar-based

concept representations (e.g., Fuzzy and Bool CBMs), will be forced to compress the information from the input data at concept level, leading to a compromise between the $I(\mathcal{X}, \hat{\mathcal{C}})$ and $I(\hat{\mathcal{C}}, \mathcal{Y})$.

Following the approach of (Kolchinsky et al., 2019; Saxe et al., 2018) we approximate the Mutual Information (MI) through the Kernel Density Estimation (KDE) method. Kolchinsky et al. (2019) show that this method accurately approximates the MI computed through the binning procedure proposed by Tishby et al. (2000). The KDE approach assumes that the activity of the analysed layer (in this case, the concept encoding layer $\hat{\mathcal{C}}$) is distributed as a mixture of Gaussians. This approximation holds true if the input samples used for evaluation are representative of the true input distribution. Therefore, we can consider the input distribution as delta functions over each sample in the dataset. Moreover, Gaussian noise is added to the layer activity to bound the mutual information w.r.t. the input – i.e., $\hat{c} = \hat{\mathbf{c}} + \epsilon$, where $\hat{\mathbf{c}}$ is the bottleneck activation vector and $\epsilon \sim N(0, \sigma^2 I)$ is a noise matrix with noise variance σ^2. In this setting, the KDE estimation of the MI with the input is:

$$I(\hat{\mathcal{C}}; \mathcal{X}) = H(\hat{\mathcal{C}}) - H(\hat{\mathcal{C}}|\mathcal{X}) = H(\hat{\mathcal{C}}) \leq \frac{\zeta}{2} - \frac{1}{n} \sum_{i=1}^{n} \log \left(\frac{1}{n} \frac{1}{2\pi\sigma^2} \sum_{j=1}^{n} e^{\frac{||\hat{c}^{(i)} - \hat{c}^{(j)}||_2^2}{2\sigma^2}} \right), \quad (4.5)$$

where n is the number of input samples and ζ is the dimension of the concept encoding layer $\hat{\mathcal{C}}$ (e.g., $\zeta = m \cdot k$ for CEM). Notice that Shwartz-Ziv and Tishby (2017) neglect the conditional entropy term arguing that the output of any neural network layer is a deterministic function of the input, which implies $H(\hat{\mathcal{C}}|\mathcal{X}) = 0$.

When considering instead the mutual information w.r.t. the downstream task label distribution $\mathcal{Y}$, the conditional entropy is $H(\hat{\mathcal{C}}|\mathcal{Y}) \neq 0$ and the mutual information $I(\hat{\mathcal{C}}; \mathcal{Y})$ can be estimated as:

$$\begin{aligned} I(\hat{\mathcal{C}}; \mathcal{Y}) = H(\hat{\mathcal{C}}) - H(\hat{\mathcal{C}}|\mathcal{Y}) \leq \frac{\zeta}{2} - \frac{1}{n} \sum_{i=1}^{n} \log \left(\frac{1}{n} \frac{1}{2\pi\sigma^2} \sum_{j=1}^{n} e^{\frac{||\hat{c}^{(i)} - \hat{c}^{(j)}||_2^2}{2\sigma^2}} \right) \\ - \sum_{l=1}^{L} p_l \left[\frac{\zeta}{2} - \frac{1}{P_l} \sum_{\substack{i \\ \text{s.t. } y^{(i)}=l}} \log \left(\frac{1}{P_l} \frac{1}{2\pi\sigma^2} \sum_{\substack{j \\ \text{s.t. } y^{(j)}=l}} e^{\frac{||\hat{c}^{(i)} - \hat{c}^{(j)}||_2^2}{2\sigma^2}} \right) \right], \end{aligned}$$

where L is the number of downstream task labels, P_l the number of data with output label l, and $p_l = P_l/n$ is the probability of task label l.

When considering the concept labels $\mathcal{C}$, however, the same estimation cannot be employed since it requires the labels to be mutually exclusive. While this holds true for

the task labels $\mathcal{Y}$ in the considered settings, the concepts in $\mathcal{C}$ are generally multi-labeled — i.e., more than one concept can be true when considering a single sample $\mathbf{x}^{(i)}$. Therefore, in this case we compute the average of the conditional entropies $H(\hat{\mathcal{C}}|\mathcal{C}) = 1/k \sum_a H(\hat{\mathcal{C}}|\mathcal{C}_a)$ across all k concepts. More precisely,

$$
\begin{aligned}
I(\hat{\mathcal{C}};\mathcal{C}) &= H(\hat{\mathcal{C}}) - H(\hat{\mathcal{C}}|\mathcal{C}) \\
&= H(\hat{\mathcal{C}}) - \frac{1}{k}\sum_{a=1}^{k} H(\hat{\mathcal{C}}|\mathcal{C}_a) \\
&\leq \frac{\zeta}{2} - \frac{1}{n}\sum_{i=1}^{n} \log\left(\frac{1}{n}\frac{1}{2\pi\sigma^2}\sum_{j=1}^{n} e^{\frac{||\hat{\mathbf{c}}^{(i)} - \hat{\mathbf{c}}^{(j)}||_2^2}{2\sigma^2}}\right) \\
&\quad - \frac{1}{k}\sum_{a=1}^{k}\sum_{m\in M_a} p_{a,m}\left[\frac{\zeta}{2} - \frac{1}{P_{a,m}}\sum_{\substack{i \\ \text{s.t. } c_a^{(i)}=m}} \log\left(\frac{1}{P_{a,m}}\frac{1}{2\pi\sigma^2}\sum_{\substack{j \\ \text{s.t. } c_a^{(j)}=m}} e^{\frac{||\hat{\mathbf{c}}^{(i)} - \hat{\mathbf{c}}^{(j)}||_2^2}{2\sigma^2}}\right)\right],
\end{aligned}
$$

where $P_{a,m}$ is the number of samples having the concept $c_a = m$, M_k is the set of possible values that the c_a concept can assume (generally $M_a = \{0, 1\}$), and $p_{a,m} = P_{a,m}/n$ is the probability of concept label $c_a = m$.

In all the previous cases, since we use the natural logarithm, the MI is computed in NATS. To convert it into bits, we scale the obtained values by $\frac{1}{\log(2)}$.

The role of noise The variance σ^2 of the noise matrix ϵ, plays an important role in the computation of the MI. More precisely, low values of σ entail high negative values for $H(\hat{\mathcal{C}}|\mathcal{X})$, and, consequently, high positive values for $I(\hat{\mathcal{C}};\mathcal{X})$. In the extreme case where we do not add any noise, we have $H(\hat{\mathcal{C}}|\mathcal{X}) = -\inf$ and $I(\hat{\mathcal{C}};\mathcal{X}) \sim \inf$, as long as the entropy $H(\hat{\mathcal{C}})$ is finite. Furthermore, as we can observe in the equations above, the dimensionality ζ of the concept representation also plays an important role in the computation of the MI, the latter being directly proportional to the dimensionality of concept representation layer $\hat{\mathcal{C}}$. To mitigate this issue, we also consider the noise to be directly proportional to the dimension of $\hat{\mathcal{C}}$, by setting $\sigma^2 = \zeta/100$.

4.5 Experiments

In this section, we analyze the performance of concept embedding models by means of the following research questions:

- **Task accuracy** — What is the impact of concept embeddings on a CBM's downstream task performance? Are models based on concept embeddings still subject to an information bottleneck (Tishby et al., 2000)?

- **Explainability** — Are CEM concept-based explanations aligned with ground truth concepts? Do they offer interpretability beyond simple concept prediction and alignment?
- **Interventions** — Do CEMs allow meaningful concept interventions when compared to Hybrid or vanilla CBMs?

4.5.1 Datasets

For our evaluation, we propose three simple benchmark datasets of increasing concept complexity (from Boolean to vector-based concepts): (1) *XOR* (inspired by (Minsky and Papert, 1969)) in which tuples $(x_1, x_2) \in [0, 1]^2$ are annotated with two Boolean concepts $\{\mathbb{1}_{c_i>0.5}\}_{i=1}^2$ and labeled as $y = c_1$ XOR c_2; (2) *Trigonometric* (inspired by (Mahinpei et al., 2021)) in which three latent normal random variables $\{b_i\}_{i=1}^3$ are used to generate a 7-dimensional input whose three concept annotations are a Boolean function of $\{b_i\}_{i=1}^3$ and task label is a linear function of the same; (3) *Dot* in which four latent random vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^2$ are used to generate two concept annotations, representing whether latent vectors $\mathbf{v}_i$ point in the same direction of reference vectors $\mathbf{w}_i$, and task labels, representing whether the two latent vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ point in the same direction. Furthermore, we evaluate our methods on two real-world image tasks: the Caltech-UCSD Birds-200-2011 dataset (CUB, (Wah et al., 2011)), preprocessed as in (Koh et al., 2020), and the Large-scale CelebFaces Attributes dataset (CelebA, (Liu et al., 2015)). In our CUB task we have 112 complete concept annotations and 200 task labels while in our CelebA task we construct 6 balanced incomplete concept annotations and each image can be one of 256 classes. Therefore, we use CUB to test each model in a real-world task where concept annotations are numerous and they form a complete description of their downstream task. In contrast, our CelebA task is used to evaluate the behaviour of each method in scenarios where the concept annotations are scarce and incomplete w.r.t. their downstream task.

4.5.2 Baselines

We compare CEMs against Bool, Fuzzy, and Hybrid Joint-CBMs as they all provide concept-based explanations for their predictions and allow concept interventions at test-time. Note that this set excludes architectures such as Self-Explainable Neural Networks (Alvarez-Melis and Jaakkola, 2018) and Concept Whitening (Chen et al., 2020) as they do not offer a clear mechanism for intervening on their concept bottlenecks. To ensure fair comparison, we use the same architecture capacity across all models. Similarly, we use the same values of α and m within a dataset for all models trained on that dataset and set $p_{\text{int}} = 0.25$ when using CEM. When using Hybrid CBMs, we include as many activations in their

bottlenecks as their CEM counterparts (so that they both end up with a bottleneck with km activations) and use a Leaky-ReLU activation for unsupervised activations. Finally, in our evaluation we include a DNN without concept supervision with the same capacity as its CEM counterpart to measure the effect of concept supervision in our model's performance.

4.5.3 Metrics

We measure a model's performance based on four metrics. First, we measure task and concept classification performance in terms of both *task and mean concept accuracy*. Second, we evaluate the interpretability of learnt concept representations via our *concept alignment score*. To easily visualise the accuracy-vs-interpretability trade-off, we plot our results in a two-dimensional plane showing both task accuracy and concept alignment. Third, we study the information bottleneck in our models via *mutual information* (MI) and the Information Plane technique (Shwartz-Ziv and Tishby, 2017). Finally, we quantify user trust (Shen, 2022) by evaluating a model's task performance after concept interventions. All metrics in our evaluation, across all experiments, are computed on test sets using 5 random seeds, from which we compute a metric's mean and 95% confidence interval using the Box-Cox transformation for non-normal distributions.

4.6 Results and discussion

4.6.1 Task accuracy

MixCEM improves generalization accuracy (y-axis of Figure 4.3). Our evaluation shows that embedding-based CBMs (i.e., Hybrid-CBM and MixCEM) can achieve even better downstream accuracy than DNNs without concepts, and can easily outperform Boolean and Fuzzy CBMs by a large margin (up to +45% on Dot). This effect is emphasised when the downstream task is not a linear function of the concepts (e.g., XOR and Trigonometry) or when concept annotations are incomplete (e.g., Dot and CelebA). At the same time, we observe that all models achieve a similar high mean concept accuracy for all datasets. This suggests that, as hypothesized, the trade-off between concept accuracy and task performance in concept-incomplete tasks is significantly alleviated by the introduction of concept embeddings in a CBM's bottleneck. Finally, notice that CelebA showcases how including concept supervision during training (as in MixCEM) can lead to an even higher task accuracy than the one obtained by the no-concept model (+5%). This result further suggests that concept embedding representations enable high levels of interpretability without sacrificing performance.

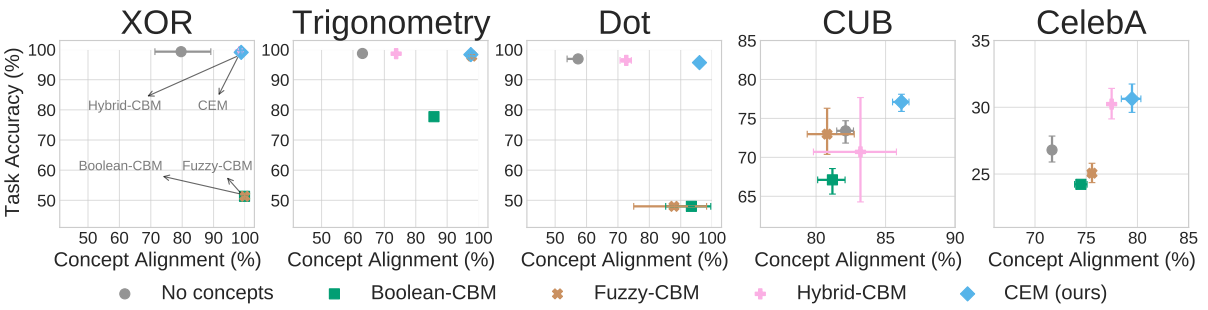


Figure 4.3: Accuracy-vs-interpretability trade-off in terms of **task accuracy** and **concept alignment score** for different concept bottleneck models. In CelebA, our most constrained task, we show the top-1 accuracy for consistency with other datasets. For these results, and those that follow, we compute all metrics on test sets across 5 seeds and report their mean and 95% confidence intervals.

MixCEM overcomes the information bottleneck (Figure 4.4). The Information Plane method indicates, as hypothesised, that embedding-based CBMs (i.e., Hybrid-CBM and MixCEM) do not compress input data information, with $I(\mathcal{X}, \mathcal{C})$ monotonically increasing during training epochs. On the other hand, Boolean and Fuzzy CBMs, as well as vanilla end-to-end models, tend to "forget" (Shwartz-Ziv and Tishby, 2017) input data information in their attempt to balance competing objective functions. Such a result constitutes a plausible explanation as to why embedding-based representations are able to maintain both high task accuracy and mean concept accuracy compared to CBMs with scalar concept representations. In fact, the extra capacity allows CBMs to maximize concept accuracy without over-constraining concept representations, thus allowing useful input information to pass by. In MixCEMs all input information flows through concepts, as they supervise the whole concept embedding. In contrast with Hybrid models, this makes the downstream tasks completely dependent on concepts, which explains the higher concept alignment scores obtained by MixCEM (see below).

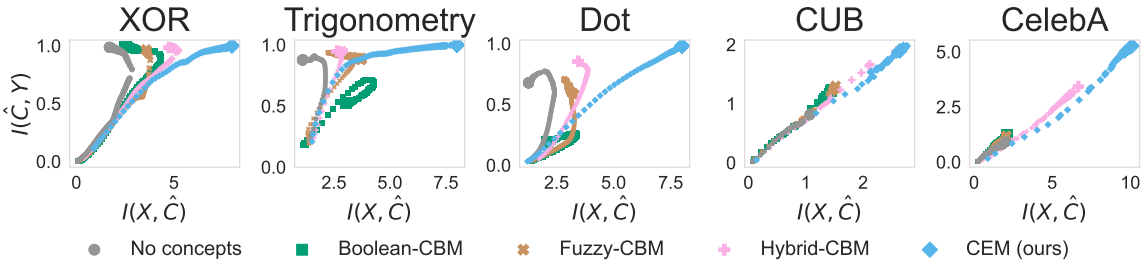


Figure 4.4: Mutual Information (MI) of concept representations ($\hat{\mathcal{C}}$) w.r.t. input distribution ($\mathcal{X}$) and ground truth labels ($\mathcal{Y}$) during training. The size of the points is proportional to the training epoch.

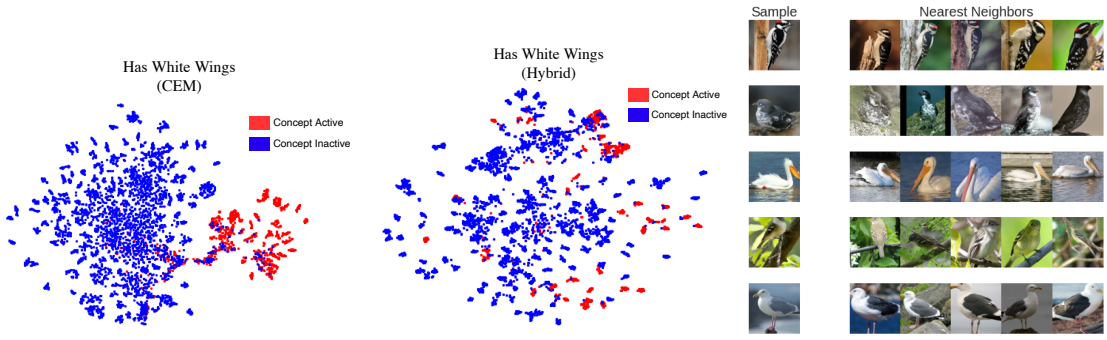
4.6.2 Explainability

MixCEM learns more interpretable concept representations (x-axis of Figure 4.3). Using the proposed CAS metric, we show that concept representations learnt by MixCEMs have alignment scores competitive or even better (e.g., on CelebA) than the ones of Boolean and Fuzzy CBMs. The alignment score also shows, as hypothesised, that hybrid concept embeddings are the least faithful representations—with alignment scores up to 25% lower than MixCEM in the Dot dataset. This is due to their unsupervised activations containing information which may not be necessarily relevant to a given concept. This result is a further evidence for why we expect interventions to be ineffective in Hybrid models (as we show shortly).

MixCEM captures meaningful concept semantics (Figure 4.5). Our concept alignment results hint at the possibility that concept embeddings learnt by MixCEM may be able to offer more than simple concept prediction. In fact, we hypothesise that their seemingly high alignment may lead to these embeddings forming more interpretable representations than Hybrid embeddings, which can lead to more useful representations for external downstream tasks. To explore this, we train a Hybrid-CBM and a MixCEM using a variation of CUB with only 25% of its concept annotations randomly selected before training, resulting in a bottleneck with 28 concepts. Once these models have been trained to convergence, we use their learnt bottleneck representations to predict the remaining 75% of the concept annotations in CUB using a simple logistic linear model. The model trained using the Hybrid bottleneck notably underperfoms when compared to the model trained using the MixCEM bottleneck (Hybrid-trained model has a mean concept accuracy of 91.83% $\pm$ 0.51% while the MixCEM-trained model's concept accuracy is 94.33% $\pm$ 0.88%). This corroborates our CAS results by suggesting that the bottlenecks learnt by MixCEMs are considerably more powerful as interpretable representations and can be used in separate downstream tasks.

We can further explore this phenomena qualitatively by visualising the embeddings learnt for a single concept using its 2-dimensional t-SNE (Van der Maaten and Hinton, 2008) plot. As shown in colour in Figure 4.5a, we can see that the embedding space learnt for a concept $\hat{\mathbf{c}}_i$ (we show here the concept "has white wings") forms two clear clusters of samples, one for points in which the concept is active and one for points in which the concept is inactive. When performing a similar analysis for the same concept in the Hybrid CBM (Figure 4.5b), where we use the entire extra capacity as the concept's representation, we see that this latent space is not as clearly separable as that in MixCEM's embeddings, suggesting this latent space is unable to capture concept-semantics as clearly as MixCEM's latent space. Notice that MixCEM's t-SNE seems to also show smaller subclusters within the activated and inactivated clusters. As Figure 4.5c shows, by looking at the nearest

Euclidean neighbours in concept's c_i embedding's space, we see that MixCEM concepts do not only clearly capture a concept's activation, but they exhibit high class-wise coherence by mapping same-type birds close to each other (explaining the observed subclusters). These results, and similar results shown in Appendix), strongly suggest that MixCEM is learning useful and interpretable high-dimensional concept representations.



(a) (b) (c)

Figure 4.5: Qualitative results: (a and b) t-SNE visualisations of "has white wings" concept embedding learnt in CUB with sample points coloured red if the concept is active in that sample, (c) top-5 test neighbours of CEM's embedding for the concept "has white wings" across 5 random test samples.

4.6.3 Interventions

MixCEM supports effective concept interventions and is more robust to incorrect interventions (Figure 4.6). When describing our MixCEM architecture, we argued in favour of using a mixture of two semantic embeddings for each concept as this would permit test-time interventions which can meaningfully affect entire concept embeddings. In Figure 4.6 left and center-left, we observe, as hypothesised, that using a mixture of embeddings allows MixCEMs to be highly responsive to random concept interventions in their bottlenecks. Notice that as predicted, although all models have a similar concept accuracy, we observe that Hybrid CBMs, while highly accurate without interventions, quickly fall short against even scalar-based CBMs once several concepts are intervened in their bottlenecks. In fact, we observe that interventions in Hybrid CBM bottlenecks have little effect on their predictive accuracy, something that did not change if logit concept probabilities were used instead of sigmoidal probabilities. More interestingly, however, we see in Figure 4.6 center-right and right that when we perform intentionally incorrect interventions (where a concept is set to the wrong value), MixCEM's performance hit is not as sharp as that of CBMs with scalar representations. We believe this is a consequence of MixCEM's "incorrect" embeddings still carrying important task-specific information which can then be used by the label predictor to produce more accurate task

labels. Finally, by comparing the effect of interventions in both MixCEMs and MixCEMs trained without RandInt, we observe that RandInt in fact leads to a model that is not just significantly more receptive to interventions, but is also able to outperform even scalar-based CBMs when large portions of their bottleneck are artificially set by experts. This suggests that our proposed architecture can not only be trusted in terms of its downstream predictions and concept explanations, as seen above, but it can also be a highly effective model when used along with experts that can correct mistakes in their concept predictions.

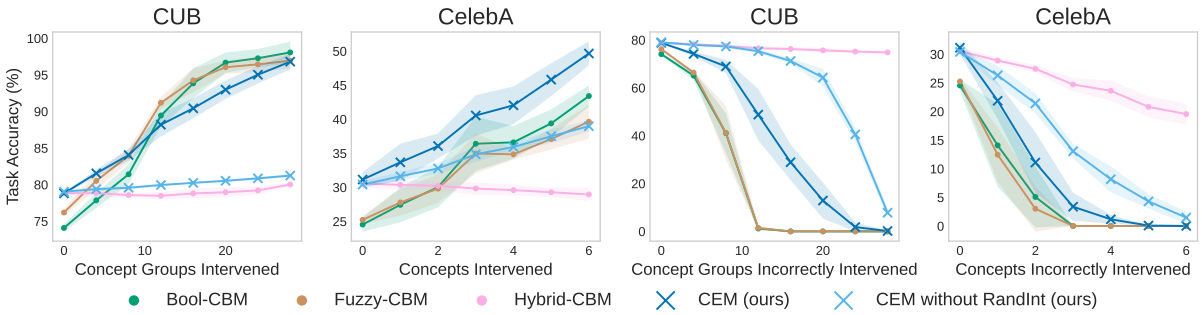


Figure 4.6: Effects of performing positive random concept interventions (left and center left) and intentionally incorrect interventions (center right and right) for different concept representations in CUB and CelebA. As in (Koh et al., 2020), when intervening in CUB we intervene using groups of concepts which are mutually exclusive.

4.7 Key findings and limitations

Overall the results of our experiments demonstrate how CEMs can fullfill our **Aim #2** as they:

- break the accuracy-explainability trade-off of concept-based models relying on concept bottlenecks (such as LENs);
- scale CBMs to real-world conditions where concept supervisions are scarce and noisy;
- support effective and simple human interventions through the learnt concepts.

Our experiments show that all existing CBMs are limited to significant accuracy-vs-interpretability trade-offs. In this respect, our work reconciles theoretical results with empirical observations: while theoretical results suggest that explicit per-concept supervisions should improve generalization bounds (Li et al., 2018), in contrast Koh et al. (2020), Chen et al. (2020), and Mahinpei et al. (2021) empirically show how learning with intermediate concepts may impair task performance in practice. The Information Plane method (Shwartz-Ziv and Tishby, 2017) reveals that the higher generalisation error

of existing concept bottleneck models might be explained as a compression in the input information flow caused by narrow architectures of Boolean and Fuzzy CBMs. In contrast, CEM represents the first concept-based model which does not need to compromise between task accuracy, concept interpretability or intervention power, thus filling this gap in the literature.

In fact, CEMs' task accuracy is higher than equivalent black boxes and comparable with hybrid CBMs. At the same time, CEMs' concept alignment is as good as in vanilla CBMs and much higher than in hybrid or fuzzy CBMs for challenging tasks. This can be explained thanks to the high-dimensional concept representation in CEMs which allows more information to flow through concepts, thus breaking the information bottleneck at concept level. As all neurons at concept level are supervised, all the information flowing through task depends on concept neurons, which makes tasks fully dependent on concepts, making CEMs able to support efficient concept interventions which increase human trust as opposed to hybrid CBMs.

However, while CEMs represent a significant advance over CBMs, the high-dimensional representation of concept embeddings prevents a straightforward understanding of the decision process. This prevents the extraction of simple logic explanations using concept-based models such decision trees or logic explained networks. As a result, in the next chapter we focus on the design of ent-to-end neural architectures reconciling **Aim #1** and **Aim #2** i.e., neural models able to generate compound explanations in formal languages while providing state-of-the-art task accuracy.

Chapter 5

Interpretable deep concept reasoning (beyond explainability)

Motivation—As we illustrated in the previous chapter, concept embeddings are a powerful concept representation as they allow concept-based models to attain state-of-the-art task performance while keeping intact the semantics of concepts. In fact, high-dimensional embeddings allow concept encoders to incorporate additional information coming from the input space which might be specific to each sample. Concept-based models can then leverage this extra information to make task predictions more accurate by considering instance-specific conditions. However, existing concept-based models are not designed to deal efficiently with concept embeddings. In fact, existing concept-based models always assume that all their input features are semantically meaningful. This way these models can leverage concept semantics to generate meaningful explanations. However, concept embeddings break this assumption as the individual features of the embedding do not have an explicit semantics. For this reason, even the most interpretable concept-based model fails to provide meaningful explanations when applied on concept embeddings.

Solution—To fill this gap, in this chapter we introduce the Deep Concept Reasoner (DCR, (Barbiero et al., 2023a)) the first interpretable concept-based model using concept embeddings. In particular DCR represents the first differentiable concept-based model able to:

- attain state-of-the-art performance in solving complex tasks, thanks to concept embeddings;
- provide human-understandable and formal explanations for its decisions.

The **key innovation** of DCR is the use of neural networks to generate interpretable rules. In particular, DCR generates the structure of logic rules whose terms are concept literals. To build these rules, DCR uses concept embeddings which allow DCR to customize logic expressions for different input samples as embeddings may hold instance-specific

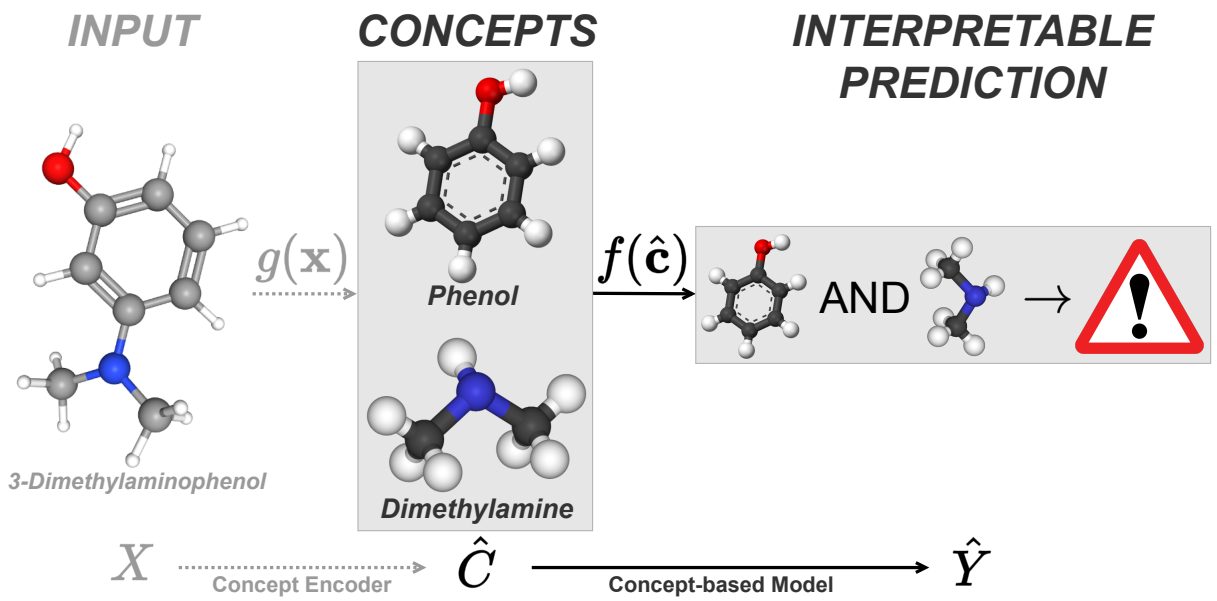


Figure 5.1: An interpretable concept-based model f maps concepts $\hat{C}$ to tasks $\hat{Y}$ generating an interpretable rule. When input features are not semantically meaningful, a concept encoder g can map raw features to a concept space.

contextual information. Having generated the logic rule, DCR then executes the logic expression evaluating concept literals on the semantically meaningful concept truth degrees.

In this chapter we will first explain how concept embedding models are not interpretable and how this may undermine user trust (Section 5.1). We will then present the syntax, the semantics and the architecture of our novel approach i.e., the Deep Concept Reasoner (Section 5.2) which represents the first interpretable concept embedding model. We will then describe the experimental setup (Section 5.3) and demonstrate how DCR outperforms existing interpretable models while provide simple logic explanations (Section 5.4). Finally we will discuss how DCR and concept embeddings represent a significant innovation in the context of explainable and neural-symbolic AI (Section 5.5).

5.1 What is wrong with explainable models?

The notions of *interpretability* and *explainability* were historically used as synonyms. However the distinction between interpretable models and models providing explanations has now become more evident, as recently discussed by different authors (Gilpin et al., 2018; Lipton, 2018; Marcinkevičs and Vogt, 2020). Even if there are no common accepted formal definitions, a model is considered interpretable when its decision process is generally transparent and can be understood directly by its structure and parameters, such as linear models or decision trees. On the other hand, the way an existing (black-box) model makes predictions can be explained by a surrogate interpretable model or by means of techniques

providing intelligible descriptions of the model behaviour e.g., logic explained networks, saliency maps, question-answering. In some contexts, the use of a black-box model may be unnecessary or even not preferable (Doshi-Velez and Kim, 2017, 2018; Ahmad et al., 2018; Rudin, 2019; Samek et al., 2020; Rudin et al., 2021). For instance, a proof of concept, a prototype, or the solution to a simple classification problem can be easily based on standard interpretable by design AI solutions (Breiman et al., 1984; Schmidt and Lipson, 2009; Letham et al., 2015; Cranmer et al., 2019; Molnar, 2020). However, interpretable models may generally miss to capture complex relationships among data. Hence, in order to achieve state-of-the-art performance in more challenging problems, it may be necessary to leverage black-box models (Battaglia et al., 2018; Devlin et al., 2018; Dosovitskiy et al., 2020; Xie et al., 2020) that, in turn, may require an additional explanatory model to gain the trust of the user.

However, the main issue of explaining black-box models is that the explanations may not be perfectly faithful to the model. In a way, the extraction of explanations often requires a form of model simplification which may be significantly misleading for human users. This misalignment between explanations and actual model behavior is one of the main reason why explainable models have been harshly criticized, especially when they are used for high-stakes decisions (Rudin, 2019). In these contexts, interpretable models are way more robust and trustworthy as their behavior does not require further explanations, thus preventing all forms of misalignment. For this reason, the challenge we face in this chapter is so important. In fact, in the previous chapters we presented techniques (LENs and CEMs) which are extremely powerful, but they need different form of explanations which may not perfectly match the true model behavior. Here instead we focus on devising a new concept-based model which builds on top of the methods in the previous chapters while aiming to be fully interpretable.

5.2 Deep concept reasoning

Here we describe the "Deep Concept Reasoner" (DCR, Figure 5.2), the first interpretable concept-based model based on concept embeddings. Similarly to existing models based on concept embeddings, DCR exploits high dimensional representations of the concepts. However, in DCR, such representations are only used to compute a logic rule. The final prediction is then obtained by evaluating such rule on the concepts truth values and not on their embeddings, thus maintaining a clear semantics and providing a totally interpretable decision. Being differentiable, DCR is trainable as an independent module on concept databases, but it can also be trained end-to-end with differentiable concept encoders. In the following section, we describe (1) the syntax of the rules we aim to learn (Section 5.2.1), (2) how to (neurally) generate and execute learnt rules to predict task labels (Section 5.2.2),

(3) how DCR learns simple rules in specific t-norm semantics (Section 5.2.2), and (4) how we can generate global and counterfactual explanations with DCR (Section 5.2.4). We provide Figure 5.2 as a reference to graphically follow the discussion.

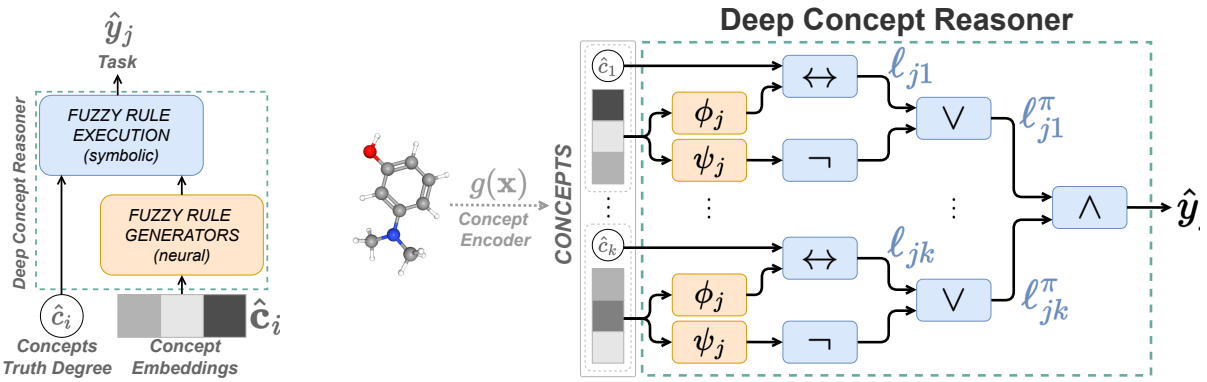


Figure 5.2: (left) Deep Concept Reasoner (DCR) generates fuzzy logic rules using neural models on concept embeddings. Then DCR executes the rule using the concept truth degrees to evaluate the rule symbolically. (right) Schema of DCR modules: first neural models ϕ and ψ generate the rule, and then the rule is executed symbolically.

5.2.1 Rule syntax

To understand the rationale behind DCR's design, we begin with an illustrative toy example:

Example 5.2.1. Consider the problem of defining the fruit "banana" given the vocabulary of concepts "soft", "round", and "yellow". A simple definition can be $y_{banana} \Leftrightarrow \neg c_{round} \wedge c_{yellow}$. From this rule we can deduce that (i) being "soft" is irrelevant for being a "banana" (indeed bananas can be both soft or hard), and (ii) being both "not round" and "yellow" is relevant to being a "banana".

As in this example, DCR rules can express whether a concept is ***relevant*** or not (e.g., "soft"), and whether a concept plays a positive (e.g., "yellow") or negative (e.g., "not round") ***role***. To formalize this description of rule syntax, we let l_{ji} denote the literal of concept c_i (i.e., $\hat{c}_i$ or $\neg\hat{c}_i$) representing the *role* of the concept i for the j-th class. Similarly, we let $r_{ji} \in \{0, 1\}$ representing whether $\hat{c}_i$ is *relevant* for predicting the class y_j. For each sample $\mathbf{x}$ and predicted class $\hat{y}_j$, DCR learns a rule with the following syntax[1]:

$$\hat{y}_j \Leftrightarrow \bigwedge_{i:\, r_{ji}=1} l_{ji} \tag{5.1}$$

Such a rule defines a logical statement for why a given sample is predicted to have label $\hat{y}_j$ using a conjunction of relevant concept literals (i.e., $\hat{c}_i$ or $\neg\hat{c}_i$).

[1]Here and in all equations we omit the explicit dependence on $\mathbf{x}$ for simplicity, i.e., we write $\hat{y}_j$ for $\hat{y}_j(\mathbf{x})$.

5.2.2 Rule generation and execution

Having defined the syntax of DCR rules, we describe how to *generate* and *execute* these rules in a differentiable way. To generate a rule we use two neural modules ϕ_j and ψ_j which determine the role and relevance of each concept, respectively. Then, we execute each rule using the concepts' truth degrees of a given sample. We split this process into three steps: (i) learning each concept's roles, (ii) learning each concept's relevance, and (iii) predicting the task using the relevant concepts.

Concept role Generation: To determine the *role* (positive/negative) of a concept, we use a feed-forward neural network $\phi_j : \mathbb{R}^m \to [0, 1]$, with m being the dimension of each concept embedding. The neural model ϕ_j takes as input a concept embedding $\hat{\mathbf{c}}_i \in \mathbb{R}^m$ and returns a soft indicator representing the role of the concept in the formula, that is, whether in literal l_{ji} the concept should appear negated (e.g., $\phi_{banana}(\hat{\mathbf{c}}_{round}) = 0$) or not (e.g., $\phi_{banana}(\hat{\mathbf{c}}_{yellow}) = 1$). Execution: When we execute the rule, we need to compute the actual truth degree of a literal l_{ji} given its role $\phi(\hat{\mathbf{c}}_i)$. We define this truth degree $\ell_{ji} \in [0, 1]$. In particular, we want to (i) forward the same truth degree of the concept, i.e. $\ell_{ji} = \hat{c}_i$, when $\phi(\hat{\mathbf{c}}_i) = 1$, and (ii) negate it, i.e. $\ell_{ji} = \neg\hat{c}_i$, when $\phi(\hat{\mathbf{c}}_i) = 0$. This behaviour can be generalized by a fuzzy equality $\Leftrightarrow$ when both ϕ_j and $\hat{c}$ are fuzzy values, i.e.:

$$\ell_{ji} = (\phi_j(\hat{\mathbf{c}}_i) \Leftrightarrow \hat{c}_i) \tag{5.2}$$

Example 5.2.2. For a given object consider $\hat{c}_{round} = 0$ and $\phi_{banana}(\hat{\mathbf{c}}_{round}) = 0$. Then we get $\ell_{banana,round} = (\phi_{banana}(\hat{\mathbf{c}}_{round}) \Leftrightarrow \hat{c}_{round}) = \neg\hat{c}_{round} = 1$. If instead we had $\phi_{banana}(\hat{\mathbf{c}}_{round}) = 1$, then $\ell_{banana,round} = (\phi_{banana}(\hat{\mathbf{c}}_{round}) \Leftrightarrow \hat{c}_{round}) = 0$.

Concept relevance. Generation: To determine the *relevance* of a concept $\hat{\mathbf{c}}_i$, we use another feed-forward neural network $\psi_j : \mathbb{R}^m \to [0, 1]$. The model ψ_j takes as input a concept embedding $\hat{\mathbf{c}}_i \in \mathbb{R}^m$ and returns a soft indicator representing the likelihood of a concept being relevant for the formula (e.g., $\psi_{banana}(\hat{\mathbf{c}}_{soft}) = 1$) or not (e.g., $\psi_{banana}(\hat{\mathbf{c}}_{yellow}) = 0$). Execution: When we execute the rule, we need to compute the truth degree of a literal given its relevance r_{ji}. We define the truth degree of a relevant literal as $\ell^r_{ji} \in [0, 1]$, where r stands for "relevant". In particular, we want to (i) filter irrelevant concepts when $\psi_j(\hat{\mathbf{c}}_i) = 0$ by setting $\ell^r_{ji} = 1$, and (ii) retain relevant literals when $\psi_j(\hat{\mathbf{c}}_i) = 1$ by setting $\ell^r_{ji} = \ell_{ji}$. This behaviour can be generalized to fuzzy values of ψ_j as follows:

$$\ell^r_{ji} = (\psi_j(\hat{\mathbf{c}}_i) \Rightarrow \ell_{ji}) = (\neg\psi_j(\hat{\mathbf{c}}_i) \vee \ell_{ji}) \tag{5.3}$$

Note that setting $\ell^r_{ji} = 1$ makes the literal l_{ji} irrelevant since "1" is neutral w.r.t. the conjunction in Equation 5.4.

Example 5.2.3. For a given object of type "banana", let the concept "soft" be irrelevant, that is $\psi_{banana}(\hat{\mathbf{c}}_{soft}) = 0$. Then we get $\ell^r_{banana,soft} = (\psi_{banana}(\hat{\mathbf{c}}_{soft}) \Rightarrow \ell_{banana,soft}) = 1$, independently from the content of $\hat{c}_{soft}$ or $\ell_{banana,soft}$. Conversely, let the concept "yellow" by relevant, that is $\psi_{banana}(\hat{\mathbf{c}}_{yellow}) = 1$, and let its concept literal be $\ell_{banana,yellow} = \hat{\mathbf{c}}_{yellow} = 1$. As a result, we get $\ell^r_{banana,yellow} = (\psi_{banana}(\hat{\mathbf{c}}_{yellow}) \Rightarrow \ell_{banana,yellow}) = 1$.

Task prediction Finally, we conjoin the relevant literals ℓ^r_{ji} to obtain the task prediction $\hat{y}_j$:

$$\hat{y}_j = \bigwedge_{i=1}^{k} \ell^r_{ji} \tag{5.4}$$

Example 5.2.4. For a given object of type "banana", consider the following truth degrees for the concepts: $\hat{c}_{soft} = 1, \hat{c}_{round} = 0, \hat{c}_{yellow} = 1$. Consider also the following values for the role and relevance for the class "banana": $\phi_{banana}(\hat{\mathbf{c}}_i) = [0, 0, 1]$ and $\psi_{banana}(\hat{\mathbf{c}}_i) = [0, 1, 1]$ for $i \in \{soft, round, yellow\}$. Then, we obtain the final prediction for class *banana* as:

$$\begin{aligned} \hat{y}_{banana} &= \textstyle\bigwedge_{i=1}^{3} (\neg\psi_{banana}(\hat{\mathbf{c}}_i) \vee (\phi_{banana}(\hat{\mathbf{c}}_i) \Leftrightarrow \hat{c}_i)) = \\ &= (1 \vee (0 \Leftrightarrow 1)) \wedge (0 \vee (0 \Leftrightarrow 0)) \wedge (0 \vee (1 \Leftrightarrow 1)) = \\ &= (1 \vee 0) \wedge (0 \vee 1) \wedge (0 \vee 1) = 1 \wedge 1 \wedge 1 = 1 \end{aligned}$$

We remark that the models ϕ_j and ψ_j: (a) generate fuzzy logic rules using concept embeddings which might hold more information than just concept truth degrees, and (b) do not depend on the number of input concepts which makes them applicable—without retraining—in testing environments where the set of concepts available differs from the set of concepts used during training. We also remark that the whole process is differentiable as the neural models ϕ_j and ψ_j are differentiable as well as the fuzzy logic operations as we will see in the next section.

5.2.3 Rule parsimony and fuzzy semantics

Rule parsimony Simple explanations and logic rules are easier to interpret for humans (Miller, 1956; Rudin, 2019). We can encode this behaviour within the DCR architecture by enforcing a certain degree of competition among concepts to make only relevant concepts survive. To this end, we design a special activation function for the neural network ψ_j rescaling the output of a log-softmax activation:

$$\gamma_{ji} = \log\left(\frac{\exp(\text{MLP}_j(\hat{\mathbf{c}}_i))}{\sum_{i'=1}^{k}\exp(\text{MLP}_j(\hat{\mathbf{c}}_{i'}))}\right) \tag{5.5}$$

$$r_{ji} = \psi_j(\hat{\mathbf{c}}_i) = \sigma\left(\gamma_{ji} - \frac{1}{k}\sum_{i'=1}^{k}\gamma_{ji'}\right) \tag{5.6}$$

This way, if the scores γ_{ji} are uniformly distributed, then we expect the network ψ_j to select half of the concepts. We can also parametrise this function by introducing a parameter $\tau \in [-\infty, \infty]$ that allows a user to bias the default behaviour of the activation function: $r_{ji} = \sigma(\gamma_{ji} - \frac{\tau}{k}\sum_{i'=1}^{k}\gamma_{ji'})$. A user can increase τ to get more relevance scores closer to 1 (more complex rules) or decrease it to get more relevance scores closer to 0 (simpler rules).

Fuzzy semantics To create a semantically valid model, we enforce the same semantic structure in all logic and neural operations. Moreover, to train our model end-to-end, we need this semantics to be differentiable in all its operations, including logic functions. Marra et al. (2020) describe a set of possible t-norm fuzzy logics which can serve the purpose. In our experiments, we use the Gödel t-norm. With this semantics, we can rewrite Equation 5.2 as:

$$\begin{aligned}\ell_{ji} &= \phi_j(\hat{\mathbf{c}}_i) \Leftrightarrow \hat{c}_i = (\phi_j(\hat{\mathbf{c}}_i) \Rightarrow \hat{c}_i) \wedge (\hat{c}_i \Rightarrow \phi_j(\hat{\mathbf{c}}_i)) = \\ &= (\neg\phi_j(\hat{\mathbf{c}}_i) \vee \hat{c}_i) \wedge (\neg\hat{c}_i \vee \phi_j(\hat{\mathbf{c}}_i)) = \\ &= \min\{\max\{1-\phi_j(\hat{\mathbf{c}}_i), \hat{c}_i\}, \max\{1-\hat{c}_i, \phi(\hat{\mathbf{c}}_i)\}\}\end{aligned}$$

and Equation 5.4 as: $\hat{y}_j = \min_{i=1}^{k}\{\max\{1-\psi_j(\hat{\mathbf{c}}_i), \ell_{ji}\}\}$

5.2.4 Global and counterfactual explanations

Interpreting global behaviour In general, DCR rules may have different weights and concepts for different samples. However, we can still globally interpret the predictions of our model without the need for an external post-hoc explainer. To this end, we collect a batch of (or all) fuzzy rules generated DCR on the training data $\mathcal{X}_{\text{train}}$. Following Barbiero et al. (2022a), we then Booleanize the collected rules and aggregate them with a global disjunction to get a single logic formula valid for all samples of class j:

$$\hat{y}_j^C = \bigvee_{\mathbf{x}\in\mathcal{X}_{\text{train}}} \hat{y}_j(\mathbf{x}) \tag{5.7}$$

This way we obtain a global overview of the decision process of our model for each class.

Counterfactual explanations Logic rules clearly reveal which concepts play a key role in a prediction. This transparency, typical of interpretable models, facilitates the extraction of simple counterfactual explanations without the need for an external algorithm as in Abid et al. (2021). In DCR we extract simple counter-examples $x^\star$ using the logic rule as guidance. Following Wachter et al. (2017), we generate counter-examples as close as possible to the original sample $|x - x^\star| < \epsilon$. In particular, Wachter et al. (2017) proposes to perturb the input features of a model starting from the most relevant features. As

the decision process depends mostly on the most relevant features, perturbing a small set of features is usually enough to find counter-examples. To this end, we first rank the concepts present in the rule according to their relevance scores. Then, starting from the most relevant concept, we invert their truth value until the prediction of the model changes. The new rule represents a counterfactual explanation for the original prediction.

5.3 Experiments

5.3.1 Research questions

In this section, we analyze the performance of deep concept reasoners by means of the following research questions:

- **Generalization** — How does DCR generalize on unseen samples compared to interpretable and neural-symbolic models? How does DCR generalize when concepts are unsupervised?
- **Interpretability** — Can DCR discover meaningful rules? Can DCR re-discover ground-truth rules? How stable are DCR rules under small perturbations of the input compared to interpretable models and local post-hoc explainers? How long does it take to extract a counterfactual explanation from DCR compared to a non-interpretable model?

5.3.2 Datasets

We investigate our research questions using six datasets spanning three of the most common data types used in deep learning: tabular, image, and graph-structured data. We use the three benchmark datasets (*XOR*, *Trigonometry*, and *Dot*) proposed by Zarlenga et al. (2022) as they capture increasingly complex concept-to-label relationships, therefore challenging concept-based models. To test the DCR's ability to re-discover ground-truth rules we use the *MNIST-Addition* dataset (Manhaeve et al., 2018), a standard benchmark for neural-symbolic systems where one aims to predict the sum of two digits from the MNIST's dataset. Furthermore, we evaluate our methods on two real-world benchmark datasets: the Large-scale CelebFaces Attributes (*CelebA*, (Liu et al., 2015)) and the *Mutagenicity* (Morris et al., 2020) dataset. In particular we define a new *CelebA* task to simulate a real-world condition of concept "shifts" where train and test concepts are correlated (e.g., "beard" and "mustaches") but do not match exactly. To this end, we split the set of *CelebA* attributes defined by Zarlenga et al. (2022) in two partially disjoint sets and use one set of attributes for training models and one for testing. Finally, we use *Mutagenicity* as a real-world scenario the concept encoder is unsupervised. As *Mutagenicity*

does not have concept annotations, we first train a graph neural network (GNN) on this dataset, and then we use the Graph Concept Explainer (GCExplainer, (Magister et al., 2021)) to extract a set of concepts from the embeddings of the trained GNN. For dataset with concept labels instead, we generate concept embeddings and truth degrees by training a Concept Embedding Model (Zarlenga et al., 2022).

5.3.3 Baselines

We compare DCR against interpretable models, such as logistic regression (Verhulst, 1845), decision trees (Breiman et al., 1984), as well as state-of-the-art black-box classifiers, such as extreme gradient boosting (XGBoost) (Chen and Guestrin, 2016), and locally-interpretable neural models, such as the Relu Net (Ciravegna et al., 2023). We train all baseline models in two different conditions mapping concepts to tasks either using concept truth degrees or using concept embeddings (baselines marked with *CT* and *CE* in figures, respecitvely). We consider interpretable only baselines trained on concept truth degrees only, as concept embeddings lack of clear semantics assigned to each dimension. However, baselines trained on concept embeddings still provide a strong reference for task accuracy w.r.t. interpretable models. On the *MNIST-Addition* dataset we compare DCR with state-of-the-art neural-symbolic baselines including: DeepProbLog (Manhaeve et al., 2018), DeepStochLog (Winters et al., 2022), Logic Tensor Networks (Badreddine et al., 2022), and Embed2Sym (Aspis et al., 2022). This is possible as the *MNIST-Addition* dataset provides access to the full set of ground-truth rules, allowing us to train these neural-symbolic systems. Finally, we compare DCR interpretability with interpretable models, such as logistic regression and decision trees, and with local post-hoc explainers, such as the Local Interpretable Model-agnostic Explanations (LIME, (Ribeiro et al., 2016a)) applied on XGBoost.

5.3.4 Metrics

We assess each model's performance and interpretability based on four criteria. First, we measure task generalization using the Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores (Hand and Till, 2001) (the higher the better). Second, we evaluate DCR interpretability by comparing the learnt logic formulae with ground-truth rules in *XOR*, *Trigonometry*, and *MNIST-Addition* datasets, and indirectly on *Mutagenicity* by checking whether the learnt rules involve concepts corresponding to functional groups known for their harmful effects, as done by Ying et al. (2019). Third, to further assess interpretability, we measure the sensitivity of the predictions under small perturbations following Yeh et al. (2019) (the lower the better). Finally, we measure how receptive our model is to extracting meaningful counterfactual examples from its rules

by computing the number of concept perturbations required to obtain a counterfactual example following Wachter et al. (2017) (the lower the better). For each metric, we report their mean and 95% Confidence Intervals (CI) on our test sets using 5 different initialization seeds.

5.4 Results and discussion

5.4.1 Task generalization

DCR outperforms interpretable models (Figure 5.3) Our experiments show that DCR generalizes significantly better than interpretable benchmarks in our most challenging datasets. This improvement peaks when concept embeddings hold more information than concept truth degrees, as in the *CelebA* and *Dot* tasks where this deficit of information is imposed byconstruction (Zarlenga et al., 2022). This grants DCR a significant advantage (up to $\sim 25\%$ improvement in ROC-AUC) over the other interpretable baselines. This phenomenon confirms the findings by Mahinpei et al. (2021) and Zarlenga et al. (2023). In particular, the concept shift in *CelebA* causes interpretable models to behave almost randomly as the set of test concept is different from the set of train concepts (despite being correlated). DCR however still generalizes well as the mechanism generating rules only depends on concept embeddings and the embeddings hold more information on the correlation between train and test concepts w.r.t. concept truth degrees. To further test this hypothesis, we compare DCR against XGBoost, decision trees (DTs), and logistic regression trained on concept embeddings. In most cases, concept embeddings allow DTs and logistic regression to improve task generalization, but the predictions of such models are no longer interpretable. In fact, even a logic rule whose terms correspond to dimensions of a concept embedding is not semantically meaningful. In contrast, DCR uses concept embeddings to assemble rules whose terms are concept truth degrees, which makes it possible to keep the rules semantically meaningful.

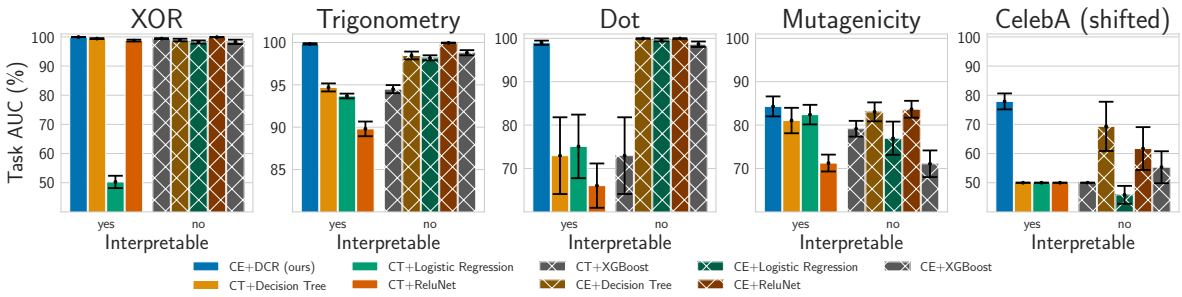


Figure 5.3: Mean ROC AUC for task predictions for all baselines across all tasks (the higher the better). DCR often outperforms interpretable concept-based models. *CE* stands for concept embeddings, while *CT* for concept truth degrees. Models trained on concept embeddings are not interpretable as concept embeddings lack a clear semantic for individual embedding dimensions.

DCR matches the accuracy of neural-symbolic systems trained using human rules (Table 5.1) Our experiments show that DCR generates rules that, when applied, obtain accuracy levels close to neural-symbolic systems trained using human rules, currently representing the gold-standard to benchmark rule learners. We show this result on the *MNIST-Addition* dataset (Manhaeve et al., 2018), a standard benchmark in neural-symbolic AI, where the labels on the concepts are not available. We learn concepts without supervision by adding another task classifier, which only uses very crisp $\hat{c}_i$ to make the task predictions. DCR achieves similar performance to state-of-the-art neural-symbolic baselines (within 1% accuracy from the best baseline). However, DCR is the only system discovering logic rules directly from data, while all the other baselines are trained using ground-truth rules. Therefore, this experiment indicates how DCR can learn meaningful rules also without concepts supervision while still maintaining state-of-the-art performance.

Table 5.1: Task accuracy on the *MNIST-addition* dataset. The neural-symbolic baselines use the knowledge of the symbolic task to distantly supervise the image recognition task. DCR achieves similar performances even though it learns the rules from scratch.

Model	**Accuracy (%)**
With ground truth rules	
DeepProbLog	97.2 ± 0.5
DeepStochLog	97.9 ± 0.1
Embed2Sym	97.7 ± 0.1
LTN	98.0 ± 0.1
Without ground truth rules	
DCR(ours)	97.4 ± 0.2

5.4.2 Interpretability

DCR discovers semantically meaningful logic rules (Table 5.2) Our experiments show that DCR induces logic rules that are both accurate in predicting the task and formally correct when compared to ground-truth logic rules. We evaluate the formal correctness of DCR rules on the *XOR*, *Trigonometry*, and *MNIST-Addition* datasets where we have access to ground-truth logic rules. We report a selection of Booleanized DCR rules with the corresponding ground truth rules in Table 5.2. Our results indicate that DCR's rules align with human-designed ground truth rules, making them highly interpretable. For instance, DCR predicts that the sum of two MNIST digits is 17 if either the first image is a  (i.e., c'_9) and the second is an  (i.e., c''_8) or vice-versa which we can interpret globally using Equation 5.7 as: $y_{17} \Leftrightarrow (c'_9 \wedge c''_8) \vee (c'_8 \wedge c''_9)$. It is interesting to investigate the potential of DCR also in settings where we do not have access to the ground-truth logic rules, such as the *Mutagenicity* dataset. Here, unlike the *MNIST addition* dataset, not only there is no supervision on the concepts, but we don't even know which are the concepts. We use GCExplainer (Magister et al., 2021) to generate a set of concepts embeddings from the embeddings of a trained GNN. We then use these embeddings to train DCR. In this setting, we can only evaluate the correctness of a DCR rules indirectly by checking whether the concepts appearing in the rules correspond to functional groups known for their harmful effects within the *Mutagenicity* dataset following Ying et al. (2019). Interestingly, many of DCR's rules predicting mutagenic effects include functional groups such as phenols (Hättenschwiler and Vitousek, 2000) and dimethylamines (ACGIH®, 2016), which can be highly toxic when combined in molecules such as 3-Dimethylaminophenols (Sabry et al., 2011). This suggests that DCR has potential to unveil semantically meaningful relations among concepts and to make them explicit to humans by means of the learnt rules.

Table 5.2: Error rate of Booleanized DCR rules w.r.t. ground truth rules. Error rate represents how often the label predicted by a Booleanised rule differs from the fuzzy rule generated by our model. The error rate is reported with the mean and standard error of the mean.

Ground-truth Rule	**Predicted Rule**	**Error (%)**
	XOR	
$y_0 \leftarrow \neg c_0 \wedge \neg c_1$	$y_0 \leftarrow \neg c_0 \wedge \neg c_1$	0.00 ± 0.00
$y_0 \leftarrow c_0 \wedge c_1$	$y_0 \leftarrow c_0 \wedge c_1$	0.00 ± 0.00
$y_1 \leftarrow \neg c_0 \wedge c_1$	$y_1 \leftarrow \neg c_0 \wedge c_1$	0.02 ± 0.02
$y_1 \leftarrow c_0 \wedge \neg c_1$	$y_1 \leftarrow c_0 \wedge \neg c_1$	0.01 ± 0.01
	Trigonometry	
$y_0 \leftarrow \neg c_0 \wedge \neg c_1 \wedge \neg c_2$	$y_0 \leftarrow \neg c_0 \wedge \neg c_1 \wedge \neg c_2$	0.00 ± 0.00
$y_1 \leftarrow c_0 \wedge c_1 \wedge c_2$	$y_1 \leftarrow c_0 \wedge c_1 \wedge c_2$	0.00 ± 0.00
	MNIST-Addition	
$y_{18} \leftarrow c'_9 \wedge c''_9$	$y_{18} \leftarrow c'_9 \wedge c''_9$	0.00 ± 0.00
$y_{17} \leftarrow c'_9 \wedge c''_8$	$y_{17} \leftarrow c'_9 \wedge c''_8$	0.00 ± 0.00
$y_{17} \leftarrow c'_8 \wedge c''_9$	$y_{17} \leftarrow c'_8 \wedge c''_9$	0.00 ± 0.00

DCR rules are stable under small perturbations (Figure 5.4) An important characteristic of local explanations is to be stable under small perturbations (Yeh et al., 2019). Indeed, users do not trust explanations if they change significantly on very similar inputs for which the model make the same prediction. This metric, also known as explanation sensitivity, is generally computed as the maximum change in the explanation of a model $\Phi(f)$ on a slightly perturbed input $(x^\star)$, that is, $|\Phi(f(\mathbf{x}^\star)) - \Phi(f(\mathbf{x}))|, |\mathbf{x} - \mathbf{x}^\star|_\infty < \epsilon$. We compare the DCR explanations w.r.t. our interpretable baselines as well as w.r.t. LIME (Ribeiro et al., 2016a) explaining the output of XGBoost. Since we are using different types of models, we use a normalised version of the sensitivity $|\Phi(f(\mathbf{x}^\star)) - \Phi(f(\mathbf{x}))|/|\Phi(f(\mathbf{x}))|$. We compute the distance between two explanations considering the feature importance of the original explanation w.r.t. to the feature importance of the explanation for the perturbed example. For decision tree's rules we consider distance between original path and the path of the perturbed example. As highlighted in Figure 5.4, in all datasets the explanations provided by DCR are very stable, particularly w.r.t. LIME and ReluNet. Notice that the figure does not report the explanation sensitivity of logistic regression and decision tree because it is trivially zero as they learn fixed rules for the entire dataset.

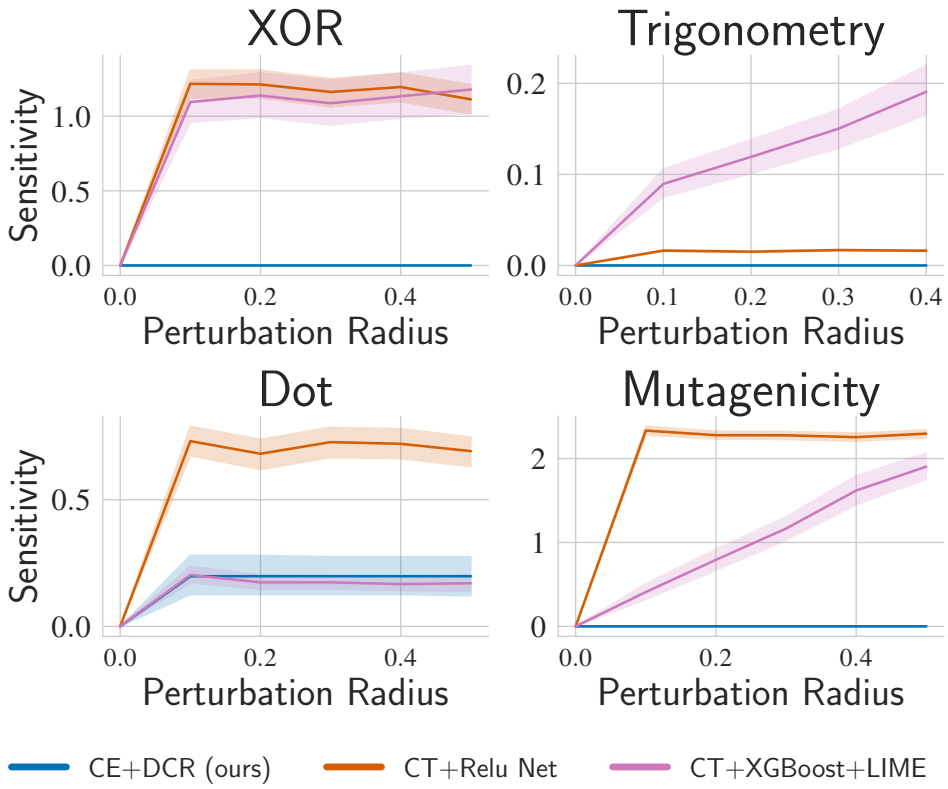


Figure 5.4: Sensitivity of model explanation when changing the radius of the input perturbation. The lower, the better. DCR explanations engender trust as they are stable under small perturbations of the input. The same does not hold generally for LIME explanations of XGBoost or Relu Net decision rules.

DCR enables discovering counterfactual examples (Figure 5.5) Besides being stable, DCR rules can be used to find simple counterfactual examples, as introduced in Section 5.2.4. In Figure 5.5 we show a model's confidence in its predictions as we increase the number of concept perturbations. In making perturbations, we sort concepts from the most relevant to the least using DCR rules, as suggested by Wachter et al. (2017). Our results show that DCR confidence in its predictions drops quickly when we perturb the most relevant concepts according to a given rule. This enables us to discover counterfactual examples where the concept literals are very similar to the original one rule. This behaviour is emblematic of interpretable models such as decision trees and logistic regression, for which similar conclusions can be drawn. We also observe how in *Mutagenicity* DCR confidence is a bit higher than interpretable baselines. We can explain this behavior as for this challenging dataset DCR rules give equal relevance to a larger set of concepts. Still DCR confidence is much lower than a black box such as XGBoost. Local explainers such as LIME can only partially explain the decision process of black box models such as XGBoost: LIME areas under the model confidence curve are generally higher than the other methods.

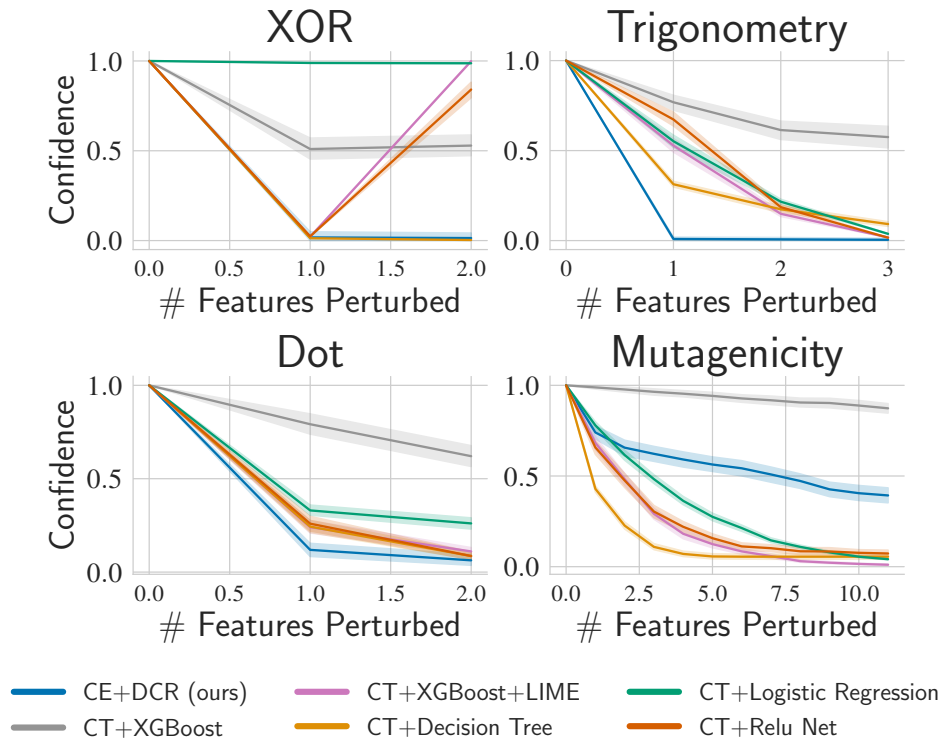


Figure 5.5: Model confidence as a function of the number of perturbed features on counterfactual examples. The lower, the better. Similarly to interpretable methods, DCR prediction confidence quickly drops after inverting the truth degree of a small set of relevant concepts, facilitating the discovery of counterfactual examples.

5.5 Key findings and limitations

Overall the results of our experiments demonstrate how DCR can fullfill our **Aim #1** and **Aim #2** at the same time as it:

- generates compound rules in a formal language;
- attains state-of-the-art performance in solving complex tasks, thanks to concept embeddings.

This is no trivial achievement as both Zarlenga et al. (2022) and Mahinpei et al. (2021) emphasise how state-of-the-art concept-based models either struggle to efficiently solve real-world tasks (when using concept truth-values only) or they weaken their interpretability (when using concept embeddings to increase their learning capacity). This is true even when concept-based models use a simple logistic regression or decision tree to map concepts embeddings to tasks because concept embedding dimensions do not have a clear semantic meaning, and models composing such dimensions generate prediction rules that are not human interpretable. Our work solves this issue by introducing the first interpretable concept-based model that learns logic rules from concept embeddings. To achieve this,

DCR builds for each sample a weighted logic rule combining neural and symbolic algorithms on concept embeddings in a unified end-to-end differentiable system.

While the global behaviour of the model is still not directly interpretable, our results show how aggregating Boolean DCR rules provides an approximation for the global behaviour of the model which matches known ground truth relationships. As a result, our experiments indicate that DCR represents a significant advance over the current state-of-the-art of interpretable concept-based models, and thus makes progress on a key research topic within the field of trustworthy AI.

Chapter 6

Conclusion

6.1 Summary of objectives and results

In this thesis I identify and discuss three of the main knowledge gaps in the explainable AI literature: deep learning opaqueness, the accuracy-explainability trade-off, and the lack of a theory for explainable AI and concept learning. My purpose is to provide a few initial contributions to address these knowledge gaps. To this aim, I propose the elements of the first sound theory of explainable AI and concept learning based on category theory. I then present logic explained networks trying to address deep learning opaqueness enabling the extraction of formal explanations from trained concept-based models. I then identify the main limitation of concept-based models in their struggle between high accuracy and good explanations. To solve this issue I propose concept embeddings which allow concept-based models to go beyond the current accuracy-explainability trade-off. Finally, to close the circle, I propose the deep concept reasoner which is the first interpretable model based on deep learning architectures and concept embeddings.

6.2 Summary of research questions and contributions

The first direction I analyze in this thesis explores the very core of explainable AI. In particular I investigate how to provide a sound formalization for the key notions in this research field trying to provide initial answers to questions such as "what is an explanation?", "what is understanding?", and "what is a concept?". To this aim I introduce the minimal set of categorical structures required to formalize these explainable AI notions. This allowed me to have a clear overview on the current research landscape and on the key knowledge gaps in explainable AI. Thanks to this analysis, I identify two main aims and research directions I further discuss in the rest of this dissertation.

The first aim I identify is to generate compound explanations for deep learning models using a formal language. In particular I analyze the problem of identifying the key concepts

for a given problem and how to combine them in simple compound formulae trying to find answers. To this end, I propose a sparse attention layer enabling differentiable concept-based models to identify the key concepts from a concept pool. Using this information, I illustrate how to extract simple and accurate logic explanations which unveil the decision process of deep neural networks. This process allowed logic explained networks to solve and explain classification problems without the need for external post-hoc explanations methods.

While LENs provide a good compromise between accuracy and explainability, they still struggle to attain state-of-the-art performances in solving complex tasks outperforming black-boxes in terms of task accuracy. I notice that this trade-off between accuracy and explainability is one of the major challenges in the field and affects almost all concept-based models. For this reason I propose concept embeddings which allow concept-based models to attain state-of-the-art performance together with good explanations.

Unfortunetely, while concept embeddings allow the extraction of good explanations, I could not find in the existing literature interpretable concept-based models designed to deal with concept embeddings. As existing models start using concept embeddings, they destroy concept semantics leading to poor explanations for their predictions. To solve this issue, in the last chapter of this work I propose the deep concept reasoner which represent the first differentiable concept-based model which is fully interpretable and it deals efficiently with concept embedding by design.

6.3 Limitations and open challenges

While this work contains significant contributions in the field of deep learning and explainable AI, each chapter also highlights several challenges which are yet to be solved.

For instance, Chapter 2 presents the first formalization of key explainable AI notions and the first theory-grounded taxonomy of the field. However, crystallize slippery notions such as "explanation" in a formal definition always represents a risk as their meaning is still largely debated even outside the field of AI in philosophy, epistemology, and psychology. My definitions do not aim to solve these debates, but rather to encourage explainable AI researchers to rigorously study these notions even further.

Chapter 3 presents logic explained networks which are able to solve and explain classification problems at the same time. However, the extraction of logic explanations requires symbolic input and output spaces. This constraint is the main limitation of our framework, as it narrows the range of applications down to symbolic input/output problems. In some contexts, such as computer vision, the use of LENs may require additional annotations and attribute labels to get a consistent symbolic layer of concepts.

However, recent work may partially solve this issue leading to more cost-effective concept annotations Ghorbani et al. (2019b); Kazhdan et al. (2020).

Chapter 4 presents concept embedding models which go beyond the current accuracy-explainability trade-off in concept-based models. While CEMs reduce the burden of choosing a sufficient amount of carefully selected concept annotations during training, which can be as expensive as task labels to obtain, these models still require enough concept annotations to be trained properly and to provide meaningful explanations. Also, while their accuracy-explainability trade-off is significantly better than the current state of the art, there is room for improvement in both concept alignment and task accuracy in challenging benchmarks such as CUB or CelebA, as well as in resource utilization during inference/training.

Chapter 5 presents deep concept reasoner which represents the first interpretable concept-based model working on concept embeddings. While the local behavior of the model is fully interpretable, the global behaviour of deep concept reasoners is still not directly interpretable, and it requires simplified explanations to be understood. Also, similarly to concept embedding models, deep concept reasoners are not as efficient in terms of resource utilization during inference/training as concept embeddings and reasoning introduce an additional burden to computations.

6.4 Real-world applications

The methods presented in this work hold the potential for a profound and wide-reaching impact within the deep learning community and beyond. These results have the capacity to transcend various machine learning scenarios, offering an adaptable framework that can be harnessed in diverse applications. One of the key strengths lies in their applicability across multiple data modalities, including images, text, signals, and graphs, which extends the reach of these methods to an array of fields.

Chemistry In the field of drug discovery, these methods may offer the potential to aid the prediction of molecular interactions. Consider a pharmaceutical researcher querying, "What is the binding affinity of chemical compound x with the target receptor y?". By exploiting existing knowledge in the field, a concept encoder may map the graphs representing the compounds x and y into a set of concepts (e.g., hydrogen bonding) and the Deep Concept Reasoner may provide an interpretable prediction by learning the following rule: "Hydrogen bonding AND hydrophobic interactions AND electrostatic attractions THEN binding affinity". This transparent explanation not only aids in selecting promising drug candidates but also provides insights into the molecular mechanisms, expediting drug

discovery and development.

Autonomous driving A similar approach could be applied in the domain of autonomous vehicles. Autonomous cars must make complex decisions based on inputs from various sensors, such as cameras and radar, such as, "Should I stop?". In this context, a concept encoder could be used to map multimodal inputs (e.g., images, light detectors, ultrasonic waves) to a set of objects (e.g., pedestrian) and the Deep Concept Reasoner can exploit both existing laws and available data to learn a rule such as: "Pedestrian detected AND pedestrian's movement AND safety priority THEN stop". This way, when the autonomous vehicle detects such a situation, the rule triggers the decision to stop, and all parties (engineers, drivers, pedestrians, judges) can check the reason behind the decision. This transparency is crucial for building public trust in autonomous vehicles and ensuring their safe deployment on our roads.

Healthcare and Large Language Models In healthcare settings, these methods offer profound implications, particularly in generating patient reports when coupled with large language models (LLMs, Vaswani et al. (2017)). Consider a request for generating a patient report for Mr. Smith, who presents with specific health concerns. In this scenario, a concept encoder may map signals from sensors monitoring Mr. Smith into a set of concepts (e.g., high blood pressure) and the Deep Concept Reasoner may generate a rule that considers vital health indicators: "High blood pressure AND low saturation AND high fever THEN potential infection". This rule can then be fed to a LLM as a backbone for a report in plain English such as, "Mr. Smith has high blood pressure, low oxygen saturation levels, and high fever. These key factors might indicate a potential infection or health issue that requires immediate attention". This transparent and rule-based approach not only ensures the accuracy of the generated patient report but also offers insights into the critical health parameters, facilitating timely and effective medical intervention and contributing to responsible and reliable healthcare reporting.

These are just some concrete examples demonstrating how the methods developed in this manuscript can introduce transparency, interpretability, and reliability in various real-world scenarios. By breaking down complex decision-making processes into clear and comprehensible steps, these methods empower AI to provide insights that enhance decision-making, reliability, and ethical compliance across these domains.

6.5 Potential impact

The contributions presented in this work may have a significant impact both within the explainable AI field and outside. For instance, this work presents the first formal theory of explainable AI. In particular, I formalize key notions that were still lacking a rigorous definition. I then use these definitions to propose the first theory-grounded taxonomy of the XAI literature. Through this work, I provide a first answer to the pressing need for a more sound foundation and formalism in explainable AI as advocated by the current literature Adadi and Berrada (2018); Palacio et al. (2021). While our taxonomy provides guidance to navigate the field, our formalism strengthens the reputation of explainable AI encouraging a safe and ethical deployment of AI technologies. I think that this work may contribute in paving the way for new research lines in XAI, e.g. quantitatively investigating the mutual understanding of two agents using different signatures, or even trying to agree on different explanations.

The second main source of impact of this work is represented by concept embeddings which go beyond the current accuracy-explainability trade-off. This represented one of the major concerns in the field of explainable AI, which means that concept embeddings may represent a significant change within this field allowing concept-based models to become more robust and scalable. Thanks to this contribution, explainable AI techniques may widen their adoption and impact in real-world applications where human supervisions are scarce and noisy.

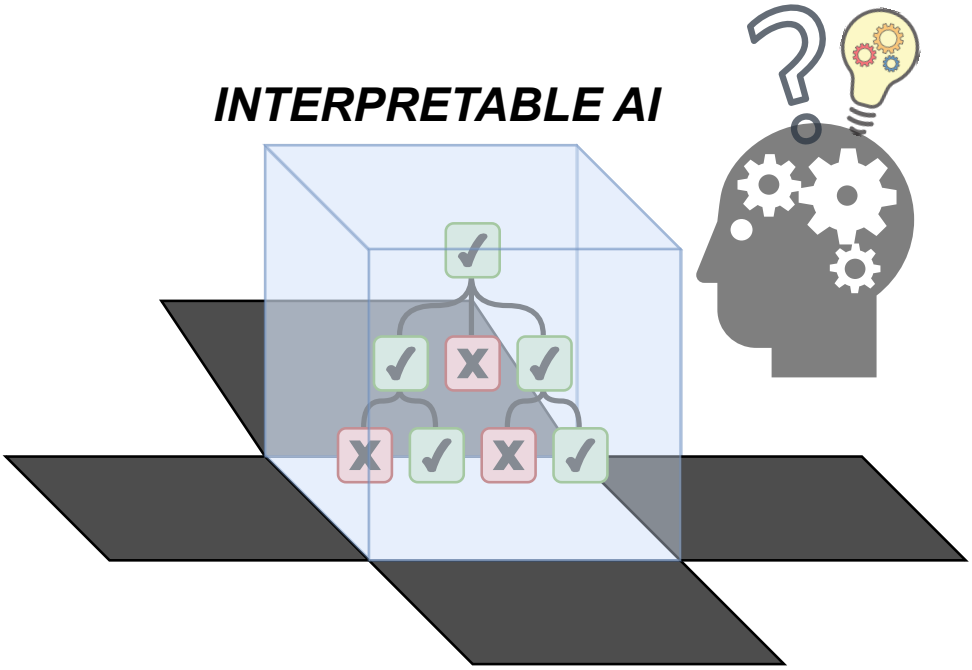


Figure 6.1: Comic representation of how the techniques presented in this work contribute in opening deep learning "black-boxes" allowing humans to understand the reasons behind predictions.

Finally, my work on logic explained networks and deep concept reasoning represent a significant advance for the lack of interpretability in deep learning systems. While deep learning is already attaining state-of-the-art performance in many fields, the lack of transparency of deep learning architectures is challenging their widespread adoption in safety-critical applications. While explainable methods represent a brittle solution to this problem, deep concept reasoning represents a fully interpretable model which attains state-of-the-art performances on challenging tasks while providing compound logic explanations for its predictions. This property can significantly improve the calibration of trust in deep learning (see Figure 6.1 for a comic representation) as it allows domain experts to understand the reasons behind deep learning predictions and to further improve the model through interventions on concept embeddings. Even more, as deep concept reasoning makes predictions using logic rules, domain experts can also interact directly with these rules leading to a new form of human-machine interaction where final decisions come from a proficuous high-level interaction where humans and machines speak a similar language.

References

Abid, A., Yuksekgonul, M., and Zou, J. (2021). Meaningfully explaining model mistakes using conceptual counterfactuals. *arXiv preprint arXiv:2106.12723*.

ACGIH® (2016). American conference of governmental industrial hygienists: Tlvs and beis based on the documentation of the threshold limit values for chemical substances and physical agents and biological exposure indices. American Conference of Governmental Industrial Hygienists Washington, DC, USA.

Adadi, A. and Berrada, M. (2018). Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160.

Ahmad, M. A., Eckert, C., and Teredesai, A. (2018). Interpretable machine learning in healthcare. In *Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics*, pages 559–560.

Alvarez-Melis, D. and Jaakkola, T. S. (2018). Towards robust interpretability with self-explaining neural networks. *arXiv preprint arXiv:1806.07538*.

Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., and Rudin, C. (2018). Learning certifiably optimal rule lists for categorical data.

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al. (2020). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115.

Aspis, Y., Broda, K., Lobo, J., and Russo, A. (2022). Embed2sym-scalable neuro-symbolic reasoning via clustered embeddings. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, volume 19, pages 421–431.

Azzolin, S., Longa, A., Barbiero, P., Liò, P., and Passerini, A. (2022). Global explainability of gnns via logic combination of learned concepts. *arXiv preprint arXiv:2210.07147 [Accepted for publication at the International Conference on Learning Representations]*.

Badreddine, S., Garcez, A. d., Serafini, L., and Spranger, M. (2022). Logic tensor networks. *Artificial Intelligence*, 303:103649.

Barbiero, P., Ciravegna, G., Giannini, F., Lió, P., Gori, M., and Melacci, S. (2022a). Entropy-based logic explanations of neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6046–6054.

Barbiero, P., Ciravegna, G., Giannini, F., Zarlenga, M. E., Magister, L. C., Tonda, A., Lio, P., Precioso, F., Jamnik, M., and Marra, G. (2023a). Interpretable neural-symbolic concept reasoning. *arXiv preprint arXiv:2304.14068 [Accepted for publication at the International Conference of Machine Learning 2023]*.

Barbiero, P., Fioravanti, S., Giannini, F., Tonda, A., Lio, P., and Di Lavore, E. (2023b). Categorical foundations of explainable ai: A unifying formalism of structures and semantics. *arXiv preprint arXiv:2304.14094*.

Barbiero, P. and Lió, P. (2020). The computational patient has diabetes and a covid. *arXiv preprint arXiv:2006.06435*.

Barbiero, P., Squillero, G., and Tonda, A. (2022b). Predictable features elimination: An unsupervised approach to feature selection. In *Machine Learning, Optimization, and Data Science: 7th International Conference, LOD 2021, Grasmere, UK, October 4–8, 2021, Revised Selected Papers, Part I*, pages 399–412. Springer.

Barbiero, P., Torné, R. V., and Lió, P. (2021). Graph representation forecasting of patient's medical conditions: Toward a digital twin. *Frontiers in genetics*, 12.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. (2018). Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.

Bredon, G. E. (2012). *Sheaf theory*, volume 170. Springer Science & Business Media.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). Classification and regression trees.

Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., and Elhadad, N. (2015). Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Chen, Z., Bei, Y., and Rudin, C. (2020). Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782.

Ciravegna, G., Barbiero, P., Giannini, F., Gori, M., Lió, P., Maggini, M., and Melacci, S. (2023). Logic explained networks. *Artificial Intelligence*, 314:103822.

Ciravegna, G., Giannini, F., Melacci, S., Maggini, M., and Gori, M. (2020). A constraint-based approach to learning and explanation. In *AAAI*, pages 3658–3665.

Coecke, B. and Kissinger, A. (2017). *Picturing Quantum Processes - A first course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press.

Coppedge, M., Gerring, J., Knutsen, C. H., Lindberg, S. I., Teorell, J., Altman, D., Bernhard, M., Cornell, A., Fish, M. S., Gastaldi, L., et al. (2021). V-dem codebook v11.

Cowan, N. (2001). The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and brain sciences*, 24(1):87–114.

Cranmer, M. D., Xu, R., Battaglia, P., and Ho, S. (2019). Learning symbolic physics with graph networks. *arXiv preprint arXiv:1909.05862*.

Cruttwell, G. S., Gavranović, B., Ghani, N., Wilson, P., and Zanasi, F. (2022). Categorical foundations of gradient-based learning. In *European Symposium on Programming*, pages 1–28. Springer, Cham.

Danilevsky, M., Qian, K., Aharonov, R., Katsis, Y., Kawas, B., and Sen, P. (2020). A survey of the state of explainable ai for natural language processing. *arXiv preprint arXiv:2010.00711*.

Das, A. and Rad, P. (2020). Opportunities and challenges in explainable artificial intelligence (xai): A survey. *ArXiv*, abs/2006.11371.

Davies, A., Veličković, P., Buesing, L., Blackwell, S., Zheng, D., Tomašev, N., Tanburn, R., Battaglia, P., Blundell, C., Juhász, A., et al. (2021). Advancing mathematics by guiding human intuition with ai. *Nature*, 600(7887):70–74.

Deasy, J., Rocheteau, E., Kohler, K., Stubbs, D. J., Barbiero, P., Lio, P., and Ercole, A. (2020). Forecasting ultra-early intensive care strain from covid-19 in england, v1. 1.4. *MedRxiv*, pages 2020–03.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Di Lavore, E., de Felice, G., and Román, M. (2022). *Monoidal Streams for Dataflow Programming*. Association for Computing Machinery, New York, NY, USA.

Di Lavore, E., Gianola, A., Román, M., Sabadini, N., and Sobociński, P. (2021). A canonical algebra of open transition systems. In *Formal Aspects of Component Software: 17th International Conference, FACS 2021, Virtual Event, October 28–29, 2021, Proceedings 17*, pages 63–81. Springer.

Di Martino, F. and Delmastro, F. (2022). Explainable ai for clinical and remote health applications: a survey on tabular and time series data. *Artificial Intelligence Review*, pages 1–55.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Doshi-Velez, F. and Kim, B. (2018). Considerations for evaluation and generalization in interpretable machine learning. In *Explainable and interpretable models in computer vision and machine learning*, pages 3–17. Springer.

Došilović, F. K., Brčić, M., and Hlupić, N. (2018). Explainable artificial intelligence: A survey. In *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 0210–0215. IEEE.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Durán, J. M. and Jongsma, K. R. (2021). Who is afraid of black box algorithms? on the epistemological and ethical basis of trust in medical ai. *Journal of Medical Ethics*, 47(5):329–335.

Eilenberg, S. and MacLane, S. (1945). General theory of natural equivalences. *Transactions of the American Mathematical Society*, 58(2):231–294.

Erhan, D., Courville, A., and Bengio, Y. (2010). Understanding representations learned in deep architectures. *Department dInformatique et Recherche Operationnelle, University of Montreal, QC, Canada, Tech. Rep*, 1355(1).

EUGDPR (2017). Gdpr. general data protection regulation.

Fox, T. (1976). Coalgebras and cartesian categories. *Communications in Algebra*, 4(7):665–667.

Fritz, T. (2020). A synthetic approach to Markov kernels, conditional independence and theorems on sufficient statistics. *Advances in Mathematics*, 370:107239.

Ganter, B. and Wille, R. (1997). Formal concept analysis: Mathematical foundations.

Georgiev, D., Barbiero, P., Kazhdan, D., Veličković, P., and Liò, P. (2022). Algorithmic concept-based explainable reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6685–6693.

Ghorbani, A., Abid, A., and Zou, J. (2019a). Interpretation of neural networks is fragile. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3681–3688.

Ghorbani, A., Wexler, J., Zou, J., and Kim, B. (2019b). Towards automatic concept-based explanations. *arXiv preprint arXiv:1902.03129*.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings.

Goguen, J. (2005). What is a concept? In *International Conference on Conceptual Structures*, pages 52–77. Springer.

Goguen, J. A. and Burstall, R. M. (1992). Institutions: Abstract model theory for specification and programming. *Journal of the ACM (JACM)*, 39(1):95–146.

Goldberger, A. L., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E. (2000). Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220.

Guidotti, R., Monreale, A., Ruggieri, S., Pedreschi, D., Turini, F., and Giannotti, F. (2018). Local rule-based explanations of black box decision systems. *arXiv preprint arXiv:1805.10820*.

Gunning, D., Stefik, M., Choi, J., Miller, T., Stumpf, S., and Yang, G.-Z. (2019). Xai—explainable artificial intelligence. *Science robotics*, 4(37):eaay7120.

Han, J., Pei, J., and Yin, Y. (2000). Mining frequent patterns without candidate generation. *ACM sigmod record*, 29(2):1–12.

Hand, D. J. and Till, R. J. (2001). A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine learning*, 45(2):171–186.

Hastie, T. and Tibshirani, R. (1987). Generalized additive models: some applications. *Journal of the American Statistical Association*, 82(398):371–386.

Hastie, T. J. (2017). Generalized additive models. In *Statistical models in S*, pages 249–307. Routledge.

Hättenschwiler, S. and Vitousek, P. M. (2000). The role of polyphenols in terrestrial ecosystem nutrient cycling. *Trends in ecology & evolution*, 15(6):238–243.

Hoffman, R. R., Mueller, S. T., Klein, G., and Litman, J. (2018). Metrics for explainable ai: Challenges and prospects. *arXiv preprint arXiv:1812.04608*.

Jain, R., Ciravegna, G., Barbiero, P., Giannini, F., Buffelli, D., and Lió, P. (2023). Extending logic explained networks to text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Jiménez-Luna, J., Grisoni, F., and Schneider, G. (2020). Drug discovery with explainable artificial intelligence. *Nature Machine Intelligence*, 2(10):573–584.

Johnstone, P. T. (2014). *Topos theory*. Courier Corporation.

Joyal, A. and Street, R. (1991). The geometry of tensor calculus, i. *Advances in mathematics*, 88(1):55–112.

Katis, P., Sabadini, N., and Walters, R. F. C. (2002). Feedback, trace and fixed-point semantics. *RAIRO-Theor. Inf. Appl.*, 36(2):181–194.

Kaufman, L. and Rousseeuw, P. J. (1990). Partitioning around medoids (program pam). *Finding groups in data: an introduction to cluster analysis*, 344:68–125.

Kazhdan, D., Dimanov, B., Jamnik, M., Liò, P., and Weller, A. (2020). Now you see me (cme): concept-based model extraction. *arXiv preprint arXiv:2010.13233*.

Kazhdan, D., Dimanov, B., Magister, L. C., Barbiero, P., Jamnik, M., and Lio, P. (2023). Gci: A (g)raph (c)oncept (i)nterpretation framework. *arXiv preprint arXiv:2302.04899*.

Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., et al. (2018). Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR.

Koh, P. W., Nguyen, T., Tang, Y. S., Mussmann, S., Pierson, E., Kim, B., and Liang, P. (2020). Concept bottleneck models. In *International Conference on Machine Learning*, pages 5338–5348. PMLR.

Kolchinsky, A., Tracey, B. D., and Wolpert, D. H. (2019). Nonlinear information bottleneck. *Entropy*, 21(12):1181.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

Kukačka, J., Golkov, V., and Cremers, D. (2017). Regularization for deep learning: A taxonomy. *arXiv preprint arXiv:1710.10686*.

Kulkarni, A., Shivananda, A., and Sharma, N. R. (2022). Explainable ai for computer vision. In *Computer Vision Projects with PyTorch*, pages 325–340. Springer.

LeCun, Y. (1998). The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

Letham, B., Rudin, C., McCormick, T. H., Madigan, D., et al. (2015). Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *Annals of Applied Statistics*, 9(3):1350–1371.

Li, C., Zia, M. Z., Tran, Q.-H., Yu, X., Hager, G. D., and Chandraker, M. (2018). Deep supervision with intermediate concepts. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1828–1843.

Li, Y., Zhou, J., Verma, S., and Chen, F. (2022). A survey of explainable graph neural networks: Taxonomy and evaluation metrics. *arXiv preprint arXiv:2207.12599*.

Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

Liu, Z., Luo, P., Wang, X., and Tang, X. (2015). Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738.

Lo Piano, S. (2020). Ethical principles in machine learning and artificial intelligence: cases from the field and possible ways forward. *Humanities and Social Sciences Communications*, 7(1):1–7.

Lopez-Rincon, A., Kidwai, S., Barbiero, P., Meijerman, I., Tonda, A., Lio, P., Maitland-van der Zee, A.-H., Oberski, D., Kraneveld, A., et al. (2022). A robust mrna signature obtained via recursive ensemble feature selection predicts the responsiveness of omalizumab in moderate-to-severe asthma. *Authorea Preprints*.

Lou, Y., Caruana, R., and Gehrke, J. (2012). Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–158.

Lundberg, S. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*.

Ma, W. J., Husain, M., and Bays, P. M. (2014). Changing concepts of working memory. *Nature neuroscience*, 17(3):347.

Mac Lane, S. (1978). *Categories for the Working Mathematician*. Graduate Texts in Mathematics. Springer New York.

Magister, L. C., Barbiero, P., Kazhdna, D., Siciliano, F., Ciravegna, G., Silvestri, F., Liò, P., and Jamnik, M. (2022). Encoding concepts in graph neural networks. *Advances in neural information processing systems*. [Under review].

Magister, L. C., Kazhdan, D., Singh, V., and Liò, P. (2021). Gcexplainer: Human-in-the-loop concept-based explanations for graph neural networks. *arXiv preprint arXiv:2107.11889*.

Mahinpei, A., Clark, J., Lage, I., Doshi-Velez, F., and Pan, W. (2021). Promises and pitfalls of black-box concept learning models. *arXiv preprint arXiv:2106.13314*.

Manhaeve, R., Dumancic, S., Kimmig, A., Demeester, T., and De Raedt, L. (2018). Deepproblog: Neural probabilistic logic programming. *Advances in Neural Information Processing Systems*, 31.

Marcinkevičs, R. and Vogt, J. E. (2020). Interpretability and explainability: A machine learning zoo mini-tour. *arXiv preprint arXiv:2012.01805*.

Marcus, G., Davis, E., and Aaronson, S. (2022). A very preliminary analysis of dall-e 2. *arXiv preprint arXiv:2204.13807*.

Margolis, E. and Laurence, S. (2007). The ontology of concepts-abstract objects or mental representations? *Noûs*, 41(4):561–593.

Marler, R. T. and Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395.

Marra, G., Giannini, F., Diligenti, M., and Gori, M. (2020). Lyrics: A general interface layer to integrate logic inference and deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 283–298. Springer.

McCluskey, E. J. (1956). Minimization of boolean functions. *The Bell System Technical Journal*, 35(6):1417–1444.

McColl, H. (1878). The calculus of equivalent statements (third paper). *Proceedings of the London Mathematical Society*, 1(1):16–28.

McKelvey, R. D. and Zavoina, W. (1975). A statistical model for the analysis of ordinal level dependent variables. *Journal of Mathematical Sociology*, 4(1):103–120.

Mendelson, E. (2009). *Introduction to mathematical logic*. CRC press.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63:81–97.

Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38.

Minsky, M. and Papert, S. A. (1969). *Perceptrons: An introduction to computational geometry*. MIT press.

Molnar, C. (2020). *Interpretable machine learning*. Lulu. com.

Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., and Neumann, M. (2020). Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*.

Ong, E. and Veličković, P. (2022). Learnable commutative monoids for graph neural networks. *arXiv preprint arXiv:2212.08541*.

Palacio, S., Lucieri, A., Munir, M., Ahmed, S., Hees, J., and Dengel, A. (2021). Xai handbook: Towards a unified framework for explainable ai. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3766–3775.

Pemstein, D., Marquardt, K. L., Tzelgov, E., Wang, Y.-t., Krusell, J., and Miri, F. (2018). The v-dem measurement model: latent variable analysis for cross-national and cross-temporal expert-coded data. *V-Dem Working Paper*, 21.

Pritchard, D. (2009). Knowledge, understanding and epistemic value. *Royal institute of philosophy supplements*, 64:19–43.

Quine, W. V. (1952). The problem of simplifying truth functions. *The American mathematical monthly*, 59(8):521–531.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1):81–106.

Quinlan, J. R. (2014). *C4. 5: programs for machine learning*. Elsevier.

Rathmanner, S. and Hutter, M. (2011). A philosophical treatise of universal induction. *Entropy*, 13(6):1076–1136.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016a). " 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*, pages 1135–1144.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016b). Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Rivest, R. L. (1987). Learning decision lists. *Machine learning*, 2(3):229–246.

Rosenberg, A. and Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 410–420.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Rudin, C., Chen, C., Chen, Z., Huang, H., Semenova, L., and Zhong, C. (2021). Interpretable machine learning: Fundamental principles and 10 grand challenges. *arXiv preprint arXiv:2103.11251*.

Sabry, N. M., Mohamed, H. M., Khattab, E. S. A., Motlaq, S. S., and El-Agrody, A. M. (2011). Synthesis of 4h-chromene, coumarin, 12h-chromeno [2, 3-d] pyrimidine derivatives and some of their antimicrobial and cytotoxicity activities. *European journal of medicinal chemistry*, 46(2):765–772.

Saeed, M., Villarroel, M., Reisner, A. T., Clifford, G., Lehman, L.-W., Moody, G., Heldt, T., Kyaw, T. H., Moody, B., and Mark, R. G. (2011). Multiparameter intelligent monitoring in intensive care ii (mimic-ii): a public-access intensive care unit database. *Critical care medicine*, 39(5):952.

Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., and Müller, K.-R. (2020). Toward interpretable machine learning: Transparent deep neural networks and beyond. *arXiv preprint arXiv:2003.07631*.

Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., and Cox, D. D. (2018). On the information bottleneck theory of deep learning. In *International Conference on Learning Representations*.

Schmidt, M. and Lipson, H. (2009). Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626.

Shen, M. W. (2022). Trust in ai: Interpretability is not necessary or sufficient, while black-box interaction is necessary and sufficient. *arXiv preprint arXiv:2202.05302*.

Shwartz-Ziv, R. and Tishby, N. (2017). Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*.

Simon, H. A. (1956). Rational choice and the structure of the environment. *Psychological review*, 63(2):129.

Simon, H. A. (1957). *Models of man; social and rational*. New York: John Wiley and Sons, Inc.

Simon, H. A. (1979). Rational decision making in business organizations. *The American economic review*, 69(4):493–513.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Soklakov, A. N. (2002). Occam's razor as a formal basis for a physical theory. *Foundations of Physics Letters*, 15(2):107–135.

Sprunger, D. and Katsumata, S. (2019). Differentiable causal computations via delayed trace. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–12. IEEE.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Tarski, A. (1944). The semantic conception of truth: and the foundations of semantics. *Philosophy and phenomenological research*, 4(3):341–376.

Tavares, A. R., Avelar, P., Flach, J. M., Nicolau, M., Lamb, L. C., and Vardi, M. (2020). Understanding boolean function learnability on deep neural networks. *arXiv preprint arXiv:2009.05908*.

Tishby, N., Pereira, F. C., and Bialek, W. (2000). The information bottleneck method. *arXiv preprint physics/0004057*.

Tishby, N. and Zaslavsky, N. (2015). Deep learning and the information bottleneck principle. In *2015 IEEE information theory workshop (ITW)*, pages 1–5. IEEE.

Tjoa, E. and Guan, C. (2020). A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE transactions on neural networks and learning systems*, 32(11):4793–4813.

Uustalu, T. and Vene, V. (2005). The essence of dataflow programming. In Yi, K., editor, *Programming Languages and Systems, Third Asian Symposium, APLAS 2005, Tsukuba, Japan, November 2-5, 2005, Proceedings*, volume 3780 of *Lecture Notes in Computer Science*, pages 2–18. Springer.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Verhulst, P. F. (1845). Resherches mathematiques sur la loi d'accroissement de la population. *Nouveaux memoires de l'academie royale des sciences*, 18:1–41.

Wachter, S., Mittelstadt, B., and Russell, C. (2017). Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. (2011). The caltech-ucsd birds-200-2011 dataset.

Wilson, A. G. (2020). The case for bayesian deep learning. *arXiv preprint arXiv:2001.10995*.

Winters, T., Marra, G., Manhaeve, R., and De Raedt, L. (2022). Deepstochlog: Neural stochastic logic programming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10090–10100.

Xie, Q., Luong, M.-T., Hovy, E., and Le, Q. V. (2020). Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698.

Xuanyuan, H., Barbiero, P., Georgiev, D., Magister, L. C., and Lió, P. (2022). Global concept-based interpretability for graph neural networks via neuron analysis. *arXiv preprint arXiv:2208.10609 [Accepted for publication at AAAI conference on artificial intelligence]*.

Yeh, C.-K., Hsieh, C.-Y., Suggala, A., Inouye, D. I., and Ravikumar, P. K. (2019). On the (in) fidelity and sensitivity of explanations. *Advances in Neural Information Processing Systems*, 32.

Yeh, C.-K., Kim, B., Arik, S., Li, C.-L., Pfister, T., and Ravikumar, P. (2020). On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33:20554–20565.

Ying, Z., Bourgeois, D., You, J., Zitnik, M., and Leskovec, J. (2019). Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32.

Zarlenga, M. E., Barbiero, P., Shams, Z., Kazhdan, D., Bhatt, U., Weller, A., and Jamnik, M. (2023). Towards robust metrics for concept representation evaluation. *arXiv preprint arXiv:2301.10367 [Accepted for publication at AAAI conference on artificial intelligence]*.

Zarlenga, M. E., Pietro, B., Gabriele, C., Giuseppe, M., Giannini, F., Diligenti, M., Zohreh, S., Frederic, P., Melacci, S., Adrian, W., et al. (2022). Concept embedding models: Beyond the accuracy-explainability trade-off. In *Advances in Neural Information Processing Systems*, volume 35, pages 21400–21413. Curran Associates, Inc.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.