EXPECTATION

PERSONALIZED EXPLAINABLE ARTIFICIAL INTELLIGENCE FOR DECENTRALIZED AGENTS WITH HETEROGENEOUS KNOWLEDGE

D2.2
Scientific paper on knowledge extraction and injection [M13]

Version: 2.0
Last Update: 08/08/2023
Distribution Level: CO

Distribution levels*
PU = Public;
RE = Restricted to a group within the given consortium;
PP = Restricted to other program participants (including commission services);

Project supported by chist-era FNSNF

CO = Confidential, only for the members of the EXPECTATION Consortium (including commission services);

Project supported by

EXPECTATION

The EXPECTATION Project Consortium groups Organizations involved:

Partner Name	**Short name**	**Country**
University of Bologna / Dep. Of Computer Science and Engineering	UNIBO	Italy

Document Identity

Creation Date:	11/03/2022
Last Update:	08/08/2023

Revision History

Version	**Edition**	**Author(s)**	**Date**
1	1	G. Ciatto, F. Sabbatini, A. Agiollo, A. Omicini	07/04/2022
2	1	G. Ciatto, F. Sabbatini, A. Agiollo, A. Omicini	18/07/2022
3	1	G. Ciatto, F. Sabbatini, A. Agiollo, M. Magnini, A. Omicini	08/08/2023
Comments:	This deliverable is actually an adaptation of D2.1 to a scientific publication. We were considering to submit it to to PeerJ Computer Science first, but later we decided to submit it to ACM Computing Surveys' special issue on Trustworthy AI. There, we received a reject & resubmit request. After revising and expanding the paper, we submitted it to the main track of ACM Computing Surveys. There, the paper is currently under review.		

Project supported by

Symbolic Knowledge Extraction and Injection with Sub-symbolic Predictors: a Systematic Literature Review

Journal:	*Computing Surveys*
Manuscript ID	CSUR-2022-0532.R1
Paper:	Long Survey Paper
Date Submitted by the Author:	08-Aug-2023
Complete List of Authors:	Ciatto, Giovanni; University of Bologna, Department of Computer Science and Enginnering (DISI) Sabbatini, Federico; University of Urbino Agiollo, Andrea; University of Bologna Magnini, Matteo; University of Bologna, Department of Computer Science and Engineering Omicini, Andrea; University of Bologna, Department of Computer Science and Engineering
Computing Classification Systems:	Theory of computation~Logic, Theory of computation~Theory and algorithms for application domains~Machine learning theory, Computing methodologies~Symbolic and algebraic manipulation~Symbolic and algebraic algorithms~Hybrid symbolic-numeric methods, Computing methodologies~Artificial intelligence~Knowledge representation and reasoning

Response to Decision Letter

Symbolic Knowledge Extraction and Injection with Sub-symbolic Predictors: a Systematic Literature Review

Dear Editors,

this is our revision of the survey "Symbolic Knowledge Extraction and Injection with Sub-symbolic Predictors: a Systematic Literature Review" which was submitted to the CSUR special issue on "Trustworthy AI" on August 08, 2022. The paper went through one round of revision, and it got two major and one minor revision recommendations from the anonymous reviewers. Then, because of the high number of submissions on the special issue, the editors invited as to resubmit the paper in main track after major revision.

In the new version of the survey, we addressed the concerns of the reviewers, and updated our survey to include contributions up to 2023. This changed the total amount of surveyed methods, which is now 246 (it was 176, before). We also analysed the surveyed methods w.r.t. the presence / lack of runnable software implementations, and extended the survey accordingly on this regard.

Accordingly, in what follows, we report detailed response to the reviewers' concerns.

Reviewer 1

[Reviewer 1 provided a very long review: 8 pages long. Here, we just quote the summary the reviewer wrote at the end of their review, and we answer to that.]

- *Bring forward to the introduction section the definition/explanation of opacity, with a reference, and explain clearly what is the paper's stance for the discussion that follows.*

We edited the introduction to mention opacity since very beginning of the paper, and we now clarify that we refer to Burrell's third definition (i.e. "Opacity as the way algorithms operate at the scale of application")

- *Revise the existing discussion on interpretability to account for the current debate in this topic, and explain why you choose the interpretability route. (This can be an opportunity to link better opacity, interpretability, and explanation, leading to trustworthy AI).*

We tried our best to improve the discussion in section 2.3 (the section about XAI), to welcome the reviewer's suggestions. However, the space is limited and the discussion is long and deep. So we just provide insights and we forward the reader towards relevant readings on these discussions—we include both our papers, and the ones suggested by the reviewers.

- *Explain better the terms that would make a system's predictions more understandable.*

Here, we tried to address altogether the many recommendations from the reviews' sec. 2.2 ("specific remarks"). As a result, we did our best to link our discussion on XAI to Trustworthy AI, as

the reviewer recommended. In particular, we added the suggested references to our section 2.3. We also clarified the link among “interpretation” and computational logic, by explicitly mentioning model theory.

- *Reference better the increasing amount of surveys on explainability (see 4th paragraph of section2.2)*

We included and explicitly mention relevant surveys about XAI, and, in particular the ones recommended by the reviewer. Clearly, for the very systematic nature of SLR, we cannot consider them directly as secondary works without compromising the reproducibility of the SRL itself.

- *Distinguish whether methods are theoretical or accompanied with experimental/practical evaluations, and give Venn diagrams that classify these.*
- *Distinguish methods that are accompanied with tools and platforms that use these tools, where applicable, and provide Venn diagrams to illustrate the results.*

We did extend the survey to take software implementations into account. As explained in section 3, we now distinguish among methods coming with no software implementations, experimental code, or reusable libraries. Summaries and statistics are now reported in the paper, while the supplementary material provide details (e.g. the exact URLs of source code repositories).

- *Justify better why SKE is to be preferred from other approaches based on ML only e.g. work on Disentangled Representation Learning for interpretability.*

There is no preference of SKE over ML-only methods. The point is just that, our survey focuses on *symbolic* knowledge. This is an a-priori choice of ours, which should not be read as a statement that SKE is superior w.r.t. purely sub-symbolic methods. We are aware that many methods exist for XAI that to not rely on symbolic knowledge. For instance, LIME and SHAP are well-known methods for XAI which do not exploit symbolic knowledge, despite being very effective and widely adopted. However, as for Disentangled Representation Learning, we did not report them because they did not fit our selection criteria.

- *Give references of hybrid agent systems of the kind envisaged in the work, past and present.*

The reviewer’s comments made us understand that the prior version of the paper was too much centred on agents. That was unintended and misleading, hence we totally understand the confusion of the reviewer. For this reason, we rephrased the paper in such a way to minimise references to the multi-agents system literature.

- *Justify why the types of computational logics referred to in the work were selected*

In section 2, we provide an overview on computational logic, and then we detail notable sorts of logics. We provide more details for the types of logic which are most frequent among the surveyed methods. We chose not to report types of logic that were not exploited by any surveyed SKE/SKI method (e.g. higher-order logics).

It is unclear whether the lack of the supplementary material was due to an accidental omission, or due to a technical submission error, but this will need to be provided with the revised version.

We uploaded the supplementary material since the very first submission, and so we did for any subsequent submission. That said, we acknowledge that the lack of supplementary materials may hinder the understanding of our paper. We are sorry for the inconvenience. In the next round of revision, we will explicitly ask the editors to forward supplementary materials to all reviewers. We also thank the reviewer for providing such a detailed review, despite lacking the supplementary materials. We hope that they manage to access the supplementary material in the next round, as we provide many details about the surveyed methods in there.

Additional Questions:

- Is the information in the paper sound, factual, and accurate?: **Yes**
  - If not, please explain why: **<empty>**
- Rate how well the ideas are presented (very difficult to understand=1), very easy to understand=5): **3**
- Rate the overall quality of the writing (very poor=1), (excellent=5): **4**
- Rate the technical quality (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **4**
- Rate the relevance to significant areas of research or practice (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **4**
- Rate the general level of interest (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **4**
- Does this paper cite and use appropriate references?: **No**
  - If not, what important references are missing?: **See attached pdf.**
- Should anything be deleted from or condensed in the paper?: **No**
  - If so, please explain:
- Is the treatment of the subject complete?: **No**
  - If not, what important details/ideas/analyses are missing?: **See attached pdf.**
- Please help ACM create a more efficient time-to-publication process: Using your best judgment, what amount of copy editing do you think this paper needs?: **Moderate**
- Most ACM journal papers are researcher-oriented. Is this paper of potential interest to developers and engineers?: **Yes**

Reviewer 2

> *This survey paper is a very complete and informative work, presenting the state of the art regarding the relationship between symbolic knowledge and popular supervised machine learning techniques. This relationship is analyzed from both direction, that is, the extraction of knowledge from predictors, with impact e.g. on explainability and interpretation, and the injection of knowledge into ML predictors, e.g. to improve their classification performance or tune their behavior.*
> *The survey is well organized. The background section is very rich and informative, and it ties together a variety of topics and terminologies which would otherwise result overwhelming.*
> *The actual survey is carried out following a sound and tested methodology, which is clearly described in detail along with the results it produced.*
> *The results of the survey are presented in the form of a series of taxonomies, organizing twenty years of work in SKE and SKI.*

We thank the reviewer for their nice words.

The conclusions are relatively short. However, the results are discussed along the paper when they are presented. Perhaps key statements or keywords of observations about the findings could be highlighted to make them stand out better.

Conclusions are deliberately short: the main "findings" of the paper are distilled in the taxonomies presented in sec. 5, hence the conclusion section simply summarises it.

One minor issue: Figure 2 on page 12 is only supported by the bibliographical reference [45]. While its general meaning is clear, it seems that the axes represent scales which are not clear to the reader. In other word, what do exactly "interpretability" and "predictive performance" mean in this chart?

The figure is deliberately ambiguous. As stated in [45] and as reported in the text, the main issue with interpretability is that it is subjective, and there is no actual measure of it. Predictive performance, in turns, is a general term referring to the score of choice for assessing a ML predictor (e.g. "accuracy" for predictors exploited in classification tasks, or "mean square errors" for regression tasks).

The whole point of the discussion around the figure is to report commonsense knowledge about what ML models are considered more (or less) "interpretable" by ML experts. To better clarify what we mean, in that section we report one particular notion of algorithmic opacity (from Burrell [11]) – namely, the one we adopt in the paper – and we argue how "interpretability" is to be read as "lack of opacity".

Additional Questions:

- Is the information in the paper sound, factual, and accurate? **Yes**
- Rate how well the ideas are presented (very difficult to understand=1), very easy to understand=5): **5**
- Rate the overall quality of the writing (very poor=1), (excellent=5): **5**
- Rate the technical quality (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **5**
- Rate the relevance to significant areas of research or practice (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **5**
- Rate the general level of interest (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **5**
- Does this paper cite and use appropriate references? **Yes**
- Should anything be deleted from or condensed in the paper? **No**
- Is the treatment of the subject complete? **Yes**
- If not, what important details/ideas/analyses are missing? **<empty>**
- Please help ACM create a more efficient time-to-publication process: Using your best judgment, what amount of copy editing do you think this paper needs? **Light**
- Most ACM journal papers are researcher-oriented. Is this paper of potential interest to developers and engineers? **Maybe**

Reviewer 3

A systematic literature review (SLR) is done to find and categorize various Symbolic Knowledge Extraction (SKE) and Injection, SKI, based predictors (ML

algorithms) in the literature. Two main goals are divided into 9 research questions.

The research questions are now 10: we added one focusing on software technologies.

Related literature was collected from scientific databases that was analyzed further. The paper is well-written with a nice structure. The contents are provided in detail.

We thank the reviewer for the nice words.

However, the main limitation is the novelty. Some surveys are already present in the literature on the black box nature of ML algorithms and how users/human can interpret or properly understand the internal working of these algorithms.

As we state in the paper, we used several prior surveys one of the starting point for our literature exploration. Arguably, our survey is unprecedented in both depth and breadth: we survey more works, more in detail as demonstrated by the fact that our taxonomies are richer. Also, this is (to the best of our knowledge) the only survey jointly considering both SKE and SKI.

Moreover, I am not sure whether this paper focus is on all predictors or Decision trees (DT) and neural networks (NN). From section 2, it seems the focus is on two predictor types (DT and NN).

Our survey focuses on predictors of all sorts. In fact, the figures in section 4, and the tables in the supplementary materials report methods involving other sorts of predictors (e.g. SVM). Nevertheless, DT and NN are, by far, the most frequent ones, and this is why we detailed them in the background section. Another reason why we do so is that DT and NN are notable representatives of models which are commonly considered highly and poorly interpretable, so believe they serve as clarifying examples in sec. 2.

Some more comments/suggestions.
1. Section 1 heading should he Introduction.

Fixed

2. No details are provided for HOL in section 2.2.2. why? as it is mentioned in the first paragraph of section 2.2. Moreover, the results section shows that no SKI is present with HOL formalisms in the literature.

The “HOL” acronym is not defined in our paper, so in this response we assume it refers to “higher-order logics”. Under this assumption, the choice of citing HOL without discussing it is deliberate, and it is due to the fact that we did not find any SKE or SKI method covering HOL.

3. It is interesting that Web of Science (WoS) is not used for paper collection. Why? as WoS and Scopus are the two most popular scientific databases.

We did use WoS in the early phases of our review, however we expected it to return proper subsets of results provided by Scopus and Scholar. We believe this is very common in this field. As our expectation proved correct in practice, we didn’t report WoS any further, in order to simplify the presentation of the paper and to reduce the burden of reproducibility.

This line "For each search engine and query pair, we consider the first two pages of result" is confusing. How many papers were present in the first two pages? It would be better if authors add a table that provides the stats for the paper collected from each database.

We thank the reviewer for the nice suggestion. Sadly, there is no more room in the paper because of the page limitation (35, ref included). Maybe this detail could be reported in supplementary material?

> *The first query keywords include NN and SVM. I think this query is contradicting the third query keyword where ML is used.*

We believe there is no contradiction. According to our experience, it is quite common for researchers in the field of ML to stick one sort of predictor and refer to that instead of ML. This is why we refer to either ML in general or to specific sorts of predictors in our queries. Concerning the choice of NN and SVM, we explicitly mention them because our expectation (then confirmed by the retrieved data) was that NN and SVM are the sorts of model which would benefit more of SKE.

> *4. How this SLR is different from some previous surveys such as [6, 12, 16, 24].*

See above.

> *5. What are the main limitations and implications of this research? They should be discussed in Section 5*

We agree. In fact, we added a "Limitations" sub-section in section 5, and a paragraph in the conclusions summarising the implications.

Additional Questions:

- Is the information in the paper sound, factual, and accurate?: **Yes**
  - If not, please explain why: **<empty>**
- Rate how well the ideas are presented (very difficult to understand=1), very easy to understand=5): **4**
- Rate the overall quality of the writing (very poor=1), (excellent=5): **4**
- Rate the technical quality (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **3**
- Rate the relevance to significant areas of research or practice (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **3**
- Rate the general level of interest (very high=5), (high=4), (moderately high=3), (low=2), (very low=1): **3**
- Does this paper cite and use appropriate references?: **No**
  - If not, what important references are missing?: **Add more references particularly in sections 3 and 4.**
- Should anything be deleted from or condensed in the paper?: **No**
  - If so, please explain: **<empty>**
- Is the treatment of the subject complete?: **No**
  - If not, what important details/ideas/analyses are missing?: **See comments to authors for more details.**
- Please help ACM create a more efficient time-to-publication process: Using your best judgment, what amount of copy editing do you think this paper needs?: **Moderate**
- Most ACM journal papers are researcher-oriented. Is this paper of potential interest to developers and engineers?: **Maybe**

Symbolic Knowledge Extraction and Injection with Sub-symbolic Predictors: a Systematic Literature Review

GIOVANNI CIATTO, Dipartimento di Informatica – Scienza e Ingegneria, Alma Mater Studiorum—Università di Bologna, Italy
FEDERICO SABBATINI, Dipartimento di Scienze Pure e Applicate, Università degli Studi di Urbino Carlo Bo, Italy
ANDREA AGIOLLO, Dipartimento di Informatica – Scienza e Ingegneria, Alma Mater Studiorum—Università di Bologna, Italy
MATTEO MAGNINI, Dipartimento di Informatica – Scienza e Ingegneria, Alma Mater Studiorum—Università di Bologna, Italy
ANDREA OMICINI, Dipartimento di Informatica – Scienza e Ingegneria, Alma Mater Studiorum—Università di Bologna, Italy

In this paper we focus on the issue of opacity of sub-symbolic machine-learning predictors by promoting two complementary activities—namely, *symbolic knowledge extraction* (SKE) and *injection* (SKI) from and into sub-symbolic predictors. We consider as symbolic any language being intelligible and interpretable for both humans and computers. Accordingly, we propose general meta-models for both SKE and SKI, along with two taxonomies for the classification of SKE/SKI methods. By adopting an eXplainable AI (XAI) perspective, we highlight how such methods can be exploited to either mitigate the aforementioned opacity issue. Our taxonomies are attained by surveying and classifying existing methods from the literature, following a systematic approach, and by generalising the results of previous surveys targeting specific sub-topics of either SKE or SKI alone. More precisely, we analyse 129 methods for SKE and 117 methods for SKI, and we categorise them according to their purpose, operation, expected input/output data and predictor types. For each method, we also indicate the presence/lack of runnable software implementations. Our work may be of interest for data scientists aiming at selecting the most adequate SKE/SKI method for their needs, and also work as suggestions for researchers interested in filling the gaps of the current state of the art, as well as for developers willing to implement SKE/SKI-based technologies.

CCS Concepts: • **Theory of computation → Logic**; *Machine learning theory*; • **Computing methodologies → Hybrid symbolic-numeric methods**; *Knowledge representation and reasoning*.

Additional Key Words and Phrases: machine learning, logic, symbolic knowledge extraction, symbolic knowledge injection

Authors' addresses: Giovanni Ciatto, Dipartimento di Informatica – Scienza e Ingegneria, Alma Mater Studiorum—Università di Bologna, via dell'Università 50, Cesena, Italy, 47522, giovanni.ciatto@unibo.it; Federico Sabbatini, Dipartimento di Scienze Pure e Applicate, Università degli Studi di Urbino Carlo Bo, Via Aurelio Saffi, 2, Urbino, Italy, 61029, f.sabbatini@unibo.it; Andrea Agiollo, Dipartimento di Informatica – Scienza e Ingegneria, Alma Mater Studiorum—Università di Bologna, via dell'Università 50, Cesena, Italy, 47522, andrea.agiollo@unibo.it; Matteo Magnini, Dipartimento di Informatica – Scienza e Ingegneria, Alma Mater Studiorum—Università di Bologna, via dell'Università 50, Cesena, Italy, 47522, matteo.magnini@unibo.it; Andrea Omicini, Dipartimento di Informatica – Scienza e Ingegneria, Alma Mater Studiorum—Università di Bologna, via dell'Università 50, Cesena, Italy, 47522, andrea.omicini@unibo.it.

0360-0300/2023/8-ART $15.00
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Giovanni Ciatto, Federico Sabbatini, Andrea Agiollo, Matteo Magnini, and Andrea Omicini. 2023. Symbolic Knowledge Extraction and Injection with Sub-symbolic Predictors: a Systematic Literature Review. *ACM Comput. Surv.* 1, 1 (August 2023), 35 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In the context of artificial intelligence (AI), more and more critical applications are being developed that rely on machine learning (ML). This promotes a data-driven approach to the engineering of intelligent computational systems where hard-to-code tasks are (semi-)automatically learned from data rather than manually programmed by human developers. Tasks that can be learned this way range from text [42] to speech [41] or image recognition [60], stepping through time series forecasting, clustering, and so on. Applications are manifold, and make our life easier in many ways—e.g., via speech-to-text applications, email spam and malware filtering, customer profiling, automatic translation, virtual personal assistants, and so forth.

Learning, in particular, is automated via ML algorithms, often implying *numeric* processing of data—which in turn enables the detection of fuzzy patterns or statistically-relevant regularities in the data, which algorithms can learn to recognise. This is fundamental to support the automatic acquisition of otherwise hard-to-formalise behaviours for computational systems. However, flexibility comes at the cost of poorly-*interpretable* solutions, as state-of-the-art *sub-symbolic* predictors – such as neural networks – are often exploited behind the scenes.

These predictors are commonly characterised by opacity [10, 32], as the interplay among the complexity of the data and the algorithms they are trained upon/with makes it hard for humans to understand their behaviour. Hence, by "interpretable" we here mean that the expert human user may observe the computational system and understand its behavior. Even though the property is not always required, there exist safety-, value-, or ethic-critical applications where humans must be in full control of the computational systems supporting their decisions or aiding their actions. In those cases, the lack of interpretability is a no-go.

State-of-the-art ML systems rely on a collection of well-established data mining predictors, such as neural networks, support vector machines, decision trees, random forests, or linear models. Despite the latter sorts of predictors being often considered as interpretable in the general case, as the complexity of the problem at hand increases (e.g., dimensionality of the available data) trained predictors become more complex, hence harder to contemplate, and therefore less interpretable. Nevertheless, these mechanisms have penetrated the modern practices of data scientists because of their flexibility, and expected effectiveness—in terms of predictive performance. Unfortunately, a number of experts have empirically observed an inverse proportionality relation among interpretability and predictive performance [13, 45]. This is the reason why data-driven engineering efforts targeting critical application scenarios nowadays have to choose between predictive performance and interpretability as their priority: we call this the *interpretability-performance trade-off*.

In this paper we focus on the problem of working around the interpretability-performance trade-off. We do so by promoting two complementary activities, namely *symbolic knowledge extraction* (SKE) and *injection* (SKI) from and into sub-symbolic predictors. In both cases, "symbolic" refers to the way knowledge is represented. In particular, we consider as symbolic any language that is intelligible and interpretable for both human beings and computers. This includes a number of logic formalisms, and excludes the fixed-sized tensors of numbers commonly exploited in sub-symbolic ML.

Intuitively, SKE is the process of distilling the knowledge a sub-symbolic predictor has grasped from data into symbolic form. This can be exploited to provide explanations for otherwise poorly-interpretable sub-symbolic predictors. More generally, SKE enables the *inspection* of the sub-symbolic predictors it is applied to, making it possible for the human designer to figure out how they will behave. Conversely, SKI is the inverse process of letting a sub-symbolic predictor follow the symbolic knowledge possibly encoded by its human designers. It enables a higher degree of *control* over a sub-symbolic predictor and its behaviour, by constraining it with human-like common-sense—suitably encoded into symbolic form.

Apart from insights, notions such as SKE and SKI have rarely been described in general terms into the scientific literature—despite the multitude of methods falling under their umbrellas. Hence, the aim of this paper is to provide general definitions and descriptions of these topics, other than providing durable taxonomies for categorising present and future SKE/SKI methods. Arguably, these contributions should take into account the widest possible portion of scientific literature, so as to avoid subjectivity. Accordingly, in this paper we propose a systematic literature review (SLR) following the three-folded purpose of *(i)* collecting and categorising existing methods for SKE and SKI into clear taxonomies, *(ii)* providing a wide overview of the state of the art and technology, and *(iii)* detecting open research challenges and opportunities. In particular, we analyse 129 methods for SKE and 117 methods for SKI, classifying them according to their purpose, operation, expected input/output data and predictor types. For each method, we also probe the existence/lack of software implementations.

To the best of our knowledge, our survey is the only *systematic* work focussing on *both* SKE and SKI algorithms. Furthemore, w.r.t. other surveys on these topics, our SLR collects the greatest number of methods. In doing so, we elicit a meta-model for SKE (resp. SKI) according to which existing and future extraction (resp. injection) methods can be categorised and described. Our taxonomies may be of interest for data scientists willing to select the most adequate SKE/SKI method for their needs, and also work as suggestions for researchers interested in filling the gaps of the current state of the art, or developers willing to implement SKE or SKI software technologies.

Accordingly, the remainder of this paper is organised as follows. Section 2 shortly recalls the state of the art for machine learning, symbolic AI, and XAI, aimed at providing readers with a fast-track access to most of the concepts and terms used in the paper. Section 3 delves into the details of what we mean by SKE and SKI, and explains how this SLR is conducted: there, we declare our research questions and describe our research methodology. Then, Section 4 answers our research questions, summarising the results of the analysis of the surveyed literature. The same results are then discussed in Section 5, where major challenges and opportunities are elicited. Finally, Section 6 concludes the paper.

2 BACKGROUND

2.1 Machine Learning

A famous definition of machine learning by [39] states:

> *a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.*

This definition is very loose, as it does not specify *(i)* what are the possible tasks, *(ii)* how performance is measured in practice, *(iii)* how / when experience should be provided to tasks, *(iv)* how exactly the program is supposed learn, and *(v)* under which form learnt information are represented. Accordingly, depending on the particular ways these aspects are tackled, a categorisation of the approaches and techniques enabling software agents to learn may be drawn.

Three major approaches to ML exist: namely, *supervised*, *unsupervised*, and *reinforcement* learning. Each approach is tailored on a well-defined pool of tasks, which may, in turn, be applied in a wide range of use case scenarios. Accordingly, differences among three major approaches can be understood by looking at the sorts of tasks T they support – commonly consisting of the estimation of some unknown relation –, and how experience E is provided to the learning algorithm.

In supervised learning, the learning task consists of finding a way to approximate an unknown relation given a sampling of its items—which constitute the experience. In unsupervised learning, the learning task consists of finding the best relation for a sample of items – which constitute the experience –, following a given optimality criterion intensionally describing the target relation. In reinforcement learning, the learning task consists of letting an agent estimate optimal plans given the reward it receives whenever it reaches particular goals. There, the rewards constitutes the experience, while plans can be described as relations among the possible states of the world, the actions to be performed in those states, and the rewards the agents expects to receive from those actions.

Several practical AI problems – such as image recognition, financial and medical decision support systems – can be reduced to *supervised* ML—which can be further grouped in terms of either *classification* or *regression* problems [29, 49]. Within the scope of sub-symbolic supervised ML, a *learning algorithm* is commonly exploited to approximate the specific nature and shape of an unknown *prediction* function (or *predictor*) $\pi^* : \mathcal{X} \to \mathcal{Y}$, mapping data from an input space $\mathcal{X}$ into an output space $\mathcal{Y}$. There, common choices for both $\mathcal{X}$ and $\mathcal{Y}$ are, for instance, the set of vectors, matrices, or tensors of numbers of a given size—hence the sub-symbolic nature of the approach.

Without loss of generality, in the following we refer to items in $\mathcal{X}$ as n-dimensional vectors denoted as $\mathbf{x}$, whereas items in $\mathcal{Y}$ are m-dimensional vectors denoted as $\mathbf{y}$—despite matrices or tensors may be suitable choices as well.

To approximate function π^*, supervised learning assumes that a *learning algorithm* is in place. This algorithm computes the approximation by taking into account a number N of *examples* of the form $(\mathbf{x}_i, \mathbf{y}_i)$ such that $\mathbf{x}_i \in X \subset \mathcal{X}$, $\mathbf{y}_i \in Y \subset \mathcal{Y}$, and $|X| \equiv |Y| \equiv N$. There, the set $D = \{(\mathbf{x}_i, \mathbf{y}_i) \mid \mathbf{x}_i \in X, \mathbf{y}_i \in Y\}$ is called *training* set, and it consists of $(n + m)$-dimensional vectors. The dataset can be considered as the concatenation of two matrices, namely the $N \times n$ matrix of *input* data (X) and the $N \times m$ matrix of *expected output* data (Y). There, each $\mathbf{x}_i$ represents an instance of the input data for which the expected output value $\mathbf{y}_i \equiv \pi^*(\mathbf{x}_i)$ is known or has already been estimated. Notably, such sorts of ML problems are said to be "supervised" *because* the expected outputs Y are available. Furthermore, the function approximation task is called **regression** if the components of Y consist of continuous or numerable – i.e., *infinite* – values, **classification** if they consist of categorical – i.e., *finite* – values.

2.1.1 On the Nature of Sub-Symbolic Data. ML methods, and sub-symbolic approaches in general, represent data as (possibly multi-dimensional) *arrays* (e.g., vectors, matrices, or tensors) of real numbers, and knowledge as functions over data. This is particularly relevant as opposed to *symbolic* knowledge representation approaches, which represent data via logic formulæ (cf. Section 2.2).

In spite of the fact that numbers are technically symbols as well, we cannot consider arrays and their functions as means for symbolic knowledge representation (KR). Indeed, according to [51], to be considered as symbolic, KR approaches should *(a)* involve a set of symbols, *(b)* which can be combined (e.g., concatenated) in possibly infinite ways, following precise grammatical rules, and *(c)* where both elementary symbols and any admissible combination of them can be assigned with *meaning*—i.e., each symbol can be mapped into some entity from the domain at hand. Below, we discuss how sub-symbolic approaches most typically do not satisfy requirements *(b)* and *(c)*.

Vectors, matrices, tensors. Multi-dimensional arrays are the basic brick of sub-symbolic data representation. More formally, a D-order array consists of an ordered container of real numbers, where D denotes the amount of indices required to locate each single item into the array. We may refer to 1-order arrays as *vectors*, 2-order arrays as *matrices*, and higher-order arrays as *tensors*.

In any given sub-symbolic data-representation task leveraging upon arrays, information may be carried by both *(i)* the actual numbers contained into the array, and *(ii)* their location into the array itself. In practice, the actual dimensions $(d_1 \times \ldots \times d_D)$ of the array play a central role as well. Indeed, sub-symbolic data processing is commonly tailored on arrays of *fixed* sizes—meaning that the actual values of $d_1, \ldots, d_D$ are chosen at design time and never changed after that. This violates requirement *(b)* above, hence we define sub-symbolic KR as the task of expressing data in the form of *rigid* arrays of *numbers*.

Local vs. distributed. When data is represented in the form of numeric arrays, the whole representation may be *local* or *distributed* [51]. In local representations, each single number into the array is characterised by a well-delimited meaning—i.e., it is measuring or describing a clearly-identifiable concept from a given domain. Conversely, in distributed representations, each single item of the array is nearly meaningless, unless it is considered along with its neighbourhood—i.e., any other item which is "close" in the indexing space of the array, according to some given notion of closeness. So, while in local representations the location of each number in the array is mostly negligible, in distributed representations it is of paramount importance. Notably, distributed representations violate the aforementioned requirement *(c)*.

2.1.2 Overview on ML Predictors. Depending on the predictor family of choice, the nature of the admissible hypothesis spaces and learning algorithms may vary dramatically, as well as the predictive performance of the target predictor, and the whole efficiency of learning.

In the literature of machine learning, statistical learning, and data mining, a plethora of learning algorithms have been proposed along the years. Because of the "no free lunch" (NFL) theorem [56], however, no algorithm is guaranteed to outperform the others in all possible scenarios. For this reason, the literature and the practice of data science keeps leveraging on algorithms and methods whose first proposal was published decades ago. Most notable algorithms include, among the many others, (deep) neural networks (NN), decision trees (DT), (generalised) linear models, nearest neighbours, support vector machines (SVM), and random forests.

These algorithms can be categorised in several ways, for instance depending *(i)* on the supervised learning task they support (classification vs. regression), or *(ii)* on the underlying strategy adopted for learning (e.g., gradient descent, least squares optimisation).

Some learning algorithms (e.g., neural networks) naturally target regression problems – despite being adaptable to classification, too –, whereas others (e.g., SVM) target classification problems—while being adaptable to regression as well. Similarly, some target multi-dimensional outputs ($\mathbf{y} \in \mathbb{R}^m$, and $m > 1$), whereas others target mono-dimensional outputs ($m = 1$). Regressors are considered as the most general case, as other learning tasks can usually be defined in terms of mono-dimensional regression.

The learning strategy is inherently bound to the predictor family of choice. Neural networks, for instance, are trained via back-propagation [47] – a particular case of stochastic gradient descent (SGD), tailored on NN –, generalised linear models via Gauss' least squares method, decision trees via CART [9], etc. Even though all the aforementioned algorithms may appear interchangeable in principle – because of the NFL theorem –, their malleability is very different in practice. For instance, the least squares method involves inverting matrices of order N – where N is the amount of available examples in the training set –, making the computational complexity of learning more than quadratic in time. Furthermore, in practice, convergence of the method is not guaranteed in

the general case; instead, it is guaranteed for generalised linear models—hence it is not adopted elsewhere. Thus, learning by least squares optimisation may become impractical for big datasets or for predictor families outside the scope of generalised linear models. Conversely, the SGD method involves arbitrarily-sized subsets of the dataset (a.k.a. batches) to be processed a limited (i.e., controllable) amount of times. Hence, the complexity of SGD can be finely controlled and adapted to the computational resources at hand—e.g., by making the learning process incremental, and by avoiding all data to be loaded in memory. Moreover, SGD can be applied to several sorts of predictor families (there including neural networks and generalised linear models), as it only requires the target function to be differentiable w.r.t. its parameters. For all these reasons, despite the lack of optimality guarantees, SGD is considered as very effective, scalable, and malleable in practice, hence it is extensively exploited in the modern data science applications.

In the remainder of this subsection we focus on two families of predictors – namely, decision trees and neural networks –, and their respective learning methods. We focus precisely on them because they are related to many surveyed SKE/SKI methods. Decision trees are noteworthy because of their user friendliness, whereas neural networks are mostly popular because of their predictive performance and flexibility.

Decision trees. Decision trees are particular sorts of predictors supporting both classification and regression tasks. In their learning phase, the input space is recursively *partitioned* through a number of splits (a.k.a. *decisions*) based on the input data X, in such a way that the prediction in each partition is constant, and the error w.r.t. the expected outputs Y is minimal, while keeping the total amount of partitions low as well. The whole procedure then synthesises a number of *hierarchical* decision rules to be followed whenever the prediction corresponding to any $x \in \mathcal{X}$ must be computed. In the inference phase, decision rules are orderly evaluated from the root to some leaf, to select the portion of the input space $\mathcal{X}$ containing x. As each leaf corresponds to a single portion of the input space, the whole procedure results in a single prediction for each x.

Unlike other families of predictors, the peculiarity of decision trees lies in the particular outcome of the learning process – namely, the *tree* of decision rules – which is straightforwardly intelligible for humans and graphically representable in 2D charts. As further discussed in the remainder of this paper, this property is of paramount importance whenever the inner operation of an automatic predictor must be interpreted and understood by a human agent.

Neural networks. Neural networks are biologically-inspired computational models, made of several elementary units (neurons) interconnected into a graph (commonly, *directed* and *acyclic*, a.k.a. DAG) via *weighted* synapses. Accordingly, the most relevant aspects of NN concern the inner operation of neurons and the particular architecture of their interconnection.

Neurons are very simple numeric computational units. They accept n scalar inputs $(x_1, \ldots, x_n) = \mathbf{x} \in \mathbb{R}^n$ weighted by as many scalar weights $(w_1, \ldots, w_n) = \mathbf{w} \in \mathbb{R}^n$, and they process the linear combination $\mathbf{x} \cdot \mathbf{w}$ via an activation function $\sigma : \mathbb{R} \mapsto \mathbb{R}$, producing a scalar output $y = \sigma(\mathbf{x} \cdot \mathbf{w})$. The output of a neuron may become the input of many others, possibly forming *networks* of neurons having arbitrary topologies. These networks may be fed with any numeric information encoded as vectors of real numbers by simply letting a number of neurons produce constant outputs.

While virtually all topologies are admissible for NN, not all are convenient. Many convenient *architectures* – roughly, patterns of well-studied topologies – have been proposed into the literature [52] to serve disparate purposes—far beyond the scope of supervised machine learning. However, identification the most appropriate architecture for any given task is non-trivial: recent efforts propose to learn their construction automatically [2, 33].

Most common NN architectures are feed-forward, meaning that neurons are organised in *layers*, where neurons from layer i can only accept ingoing synapses from neurons of layers $j < i$. The

first layer is considered the input layer, which is used to *feed* the whole network, while the last one is the output layer, where predictions are drawn. In NN architectures inference lets information flow from the input to the output layer – assuming the weights of synapses are fixed –, while training lets information flow from the output to the input layer—causing the variation of weights to minimise the prediction error of the overall network.

The recent success of deep learning [20] has proved the flexibility and the predictive performance of *deep* neural networks (DNN). 'Deep' here refers to the large amount of (possibly *convolutional*) layers. In other words, DNN can learn how to apply cascades of convolutional operations to the input data. Convolutions let the network spot relevant features into the input data, at possibly different scales. This why DNN are good at solving complex pattern-recognition tasks—e.g., computer vision or speech recognition. Unfortunately, however, unprecedented predictive performances of DNN come at the cost of their increased internal complexity and greater data greediness.

2.1.3 General Supervised Learning Workflow. Briefly speaking, a ML workflow is the process of producing a suitable predictor for the available data and the learning task at hand, with the purpose of exploiting the predictor later so as to draw analyses or to drive decisions. Hence, any ML workflow is commonly described as composed of two major phases, namely training – where predictors are fitted on data – and inference—where predictors are exploited. However, in practice, further phases are included, such as data provisioning and preprocessing, as well as model selection and assessment.

In other words, before using a sub-symbolic predictor in a real-world scenario, data scientists must ensure it has been sufficiently trained and its predictive performance is sufficiently high. In turn, training requires *(i)* an adequate amount of data to be available, *(ii)* a family of predictors to be chosen (e.g., neural networks, K-nearest neighbours, linear models, etc.), *(iii)* any structural hyper–parameter to be defined (e.g., amount, type, size of layers, K, maximum order of the polynomials, etc.), *(iv)* any other learning-parameter to be fixed (e.g., learning rate, momentum, batch size, epoch limit, etc.). Data must therefore be provisioned before training, and, possibly, pre-processed to ease training itself—e.g., by normalising data or by encoding non-numeric features into numeric form. The structure of the network must be defined in terms of (roughly) input, hidden, and output layers, as well as their activation functions. Finally, hyper-parameters must be carefully tuned according to the data scientist's experience, and the time constraints and computational resources at hand.

Thus, from a coarse-grained perspective, a machine learning workflow can be conceived as composed of six major phases, and enumerated below:

(1) **sub-symbolic data gathering:** the first actual step of any ML workflow, where data is loaded in memory for later processing;
(2) **pre-processing:** the application of several bulk operations to the training data, following several purposes, such as: *(i)* homogenise the variation ranges of the many features sampled by the dataset, *(ii)* detect irrelevant features and remove them, *(iii)* construct relevant features by combining the existing ones, or *(iv)* encoding non-numeric features into numeric form;
(3) **predictor selection:** a principled search for the most adequate sort of predictor to tackle the data and the learning task at hand. This is where hyper-parameters are commonly fixed;
(4) **training:** the actual tuning of the selected predictor(s) on the available data. This is where parameters are commonly fixed;
(5) **validation:** measuring the predictive performance of trained predictors, with the purpose of assessing if and to what extent it will generalise to new, unseen data;
(6) **inference:** the final phase, where trained predictors are used to draw predictions on unknown data—i.e., different data w.r.t. the one used for training.

2.2 Computational Logic

Symbolic knowledge representation has always been regarded as a key issue since the early days of AI, as no intelligence can exist without knowledge, and no computation can occur in lack of representation. When compared to arrays of numbers, symbolic KR is far more flexible and expressive, and, in particular, more intelligible—both machine- and human-interpretable. Historically, most KR formalisms and technologies have been designed on top of *computational logic* [34], that is, the exploitation of formal logic in computer science. Consider, for instance, *deductive databases* [23], *description logics* [5], *ontologies* [17], *Horn* logic [37], *higher-order* logic [50], just to name a few.

2.2.1 Formal logics. Many kinds of logic-based KR systems have been proposed over the years, mostly relying on *first-order logic* (FOL) – either by restricting or extending it –, e.g., on description logics and modal logics, which have been used to represent, for instance, terminological knowledge and time-dependent or subjective knowledge. Here, we briefly recall the state of the art of FOL and its most relevant subsets.

First-order logic. FOL is a general-purpose logic which can be used to represent knowledge symbolically, in a very flexible way. More precisely, it allows both human and computational agents to express (i.e., write) the properties of – and the relations among – a set of entities constituting the *domain of the discourse*, via one or more *formulæ*—and, possibly, to reason over such formulæ by drawing inferences. There, the domain of the discourse $\mathbb{D}$ is the set of all relevant entities which should be represented in FOL to be amenable of formal treatment, in a particular scenario.

Informally, the syntax for the general FOL formula is defined over the assumption that there exist: *(i)* a set of *constant* or *function* symbols, *(ii)* a set of *predicate symbols*, and *(iii)* a set of *variables*. Under such assumption, a FOL formula is any expression composed of a list of quantified variables, followed by a number of *literals*, i.e., *predicates* that may or may not be prefixed by the negation operator ($\neg$). Literals are commonly combined into expressions via *logic connectives*, such as conjunction ($\wedge$), disjunction ($\vee$), implication ($\rightarrow$), or equivalence ($\leftrightarrow$).

Each predicate consists of a predicate symbol, possibly applied to one or more *terms*. Terms may be of three sorts, namely *constants*, *functions*Constants represent entities from the domain of the discourse. In particular, each constant references a different entity. Functions are combinations of one or more entities via a *function symbol*. Similarly to predicates, functions may carry one or more terms. Being containers of terms, functions enable the creation of arbitrarily complex data structures combining several elementary terms into compisite ones. Such kind of composability by recursion is what makes FOL satisfy the aforemention definition of "symbolic" valid for FOL. Finally, variables are placeholders for unknown terms—i.e., for either individual or groups of entities.

Predicates and terms are very flexible tools to represent knowledge. While terms can be used to represent or reference either entities or groups of entities from the domain of the discourse, predicates can be used to represent relations among entities, or the properties of each single entity.

Intensional vs. extensional. In logic, one may define concepts – i.e., describe data – either *extensionally* or *intensionally*. Extensional definitions are *direct* representation of data. In the particular case of FOL, this implies defining a relation or set by explicitly mentioning the entities it involves. Conversely, intensional definitions are *indirect* representations of data. In the particular case of FOL, this implies defining a relation or set by describing its elements via other relations or sets. Recursive intensional predicates are very expressive and powerful, as they enable the description of infinite sets via a finite (and commonly small) amount of formulæ—and this is one of the key benefits of FOL as a means for KR.

2.2.2 Expressiveness vs. Tractability: Notable Subsets of FOL. Tractability deals with the theoretical questions: "can a logic reasoner compute whether a logic formula is true (or not) in *reasonable* time?". Such aspects are deeply entangled with the particular reasoner of choice. Depending on which and how many features a logic includes, it may be more or less *expressive*. The higher the expressiveness, the more the complexity of the problems which may be represented via logic and processed via inference increases. This opens to the possibility, for the solver, to meet queries which cannot be answered in practical time, or, by relying upon a limited amount of memory—or, just cannot get an answer at all. Roughly speaking, more expressive logic languages make it easier for human beings to describe a particular domain – usually, requiring them to write less and more concise clauses –, at the expense of a higher difficulty for software agents to draw inferences autonomously—because of computational tractability. This is a well-understood phenomenon in both computer science and computational logic [8, 31], often referred to as the *expressiveness-tractability* trade-off.

FOL, in particular, is considered very expressive. Indeed, it comes with many undecidable, semi-decidable, or simply intractable properties. Hence, several relevant subsets of FOL have been identified into the literature, often sacrificing expressiveness for tractability. Major notions concerning these logics are recalled below.

Horn logic. Horn logic is a notable subset of FOL, characterised by a good trade-off among theoretical expressiveness and practical tractability [36].

Horn logic is designed around the notion of *Horn clause* [26]. Horn clauses are FOL formulæ having no quantifiers, and consisting of a disjunction of predicates, where only at most one literal is non-negated—or, equivalently, an implication having a single predicate as post-condition and a conjunction of predicates as pre-condition: $h \leftarrow b_1, \ldots, b_n$. There, $\leftarrow$ denotes logic implication from right to left, commas denote logic conjunction, and all b_i, as well as h, are predicates of arbitrary arity, possibly carrying FOL terms of any sort—i.e., variables, constants, or functions. Put it simply, Horn clauses are if-then rules written in reverse order, and only supporting conjunctions of predicates as pre-conditions.

Essentially, Horn logic is a very restricted subset of FOL where: *(i)* formulæ are reduced to clauses, as they can only contain predicates, conjunctions, and a single implication operator, therefore *(ii)* operators such as $\vee$, $\leftrightarrow$, or $\neg$ cannot be used, *(iii)* variables are implicitly quantified, and *(iv)* terms work as in FOL.

Datalog. Datalog is a restricted subset of FOL [3], representing knowledge via function-free Horn clauses—defined in the previous paragraph. So, essentially, Datalog is a subset of Horn logic where structured terms (i.e., recursive data structures) are forbidden. This is a direct consequence of the lack of function symbols.

Similarly to Horn logic, Datalogs's knowledge bases consist of sets of function-free Horn clauses.

Description logics (DL). Description logics are a family of subsets of FOL, generally involving some or no quantifiers, no structured terms, and no n-ary predicates such that $n \geq 3$. In other words, description logics represent knowledge by only leveraging on constants and variables, other than atomic, unary, and binary predicates.

Differences among specific variants of DL lay in which and how many logic connectives are supported, other than, of course, whether negation is supported or not. The wide variety of DL is due to the well known expressiveness–tractability trade-off. However, depending on the particular situation at hand, one may either prefer a more expressive ($\approx$ feature rich) DL variant at the price of a reduced tractability (or even decidability) of the algorithms aimed at manipulating knowledge represented through that DL, or *vice versa*.

Regardless of the particular DL variant of choice, it is common practice in the scope of DL to call *(i)* constant terms, as "individuals" – as each constant references a single entity from a given domain –, *(ii)* unary predicates, e.g., as either "classes" or "concepts" – as each predicate *groups* a set of individuals, i.e., all those individuals for which the predicate is true –, *(iii)* binary predicates, e.g., as either "properties" or "roles"—as each predicate *relates* two sets of individuals. Following such a nomenclature, any piece of knowledge can be represented in DL by tagging each relevant entity with some constant (e.g., an URL), and by defining concepts and properties accordingly.

Notably, binary predicates are of particular interest as they support connecting couples of entities altogether. This is commonly achieved via subject-predicate-object *triplets*, i.e., ground binary predicates of the form $\langle \mathsf{a}\ f\ \mathsf{b} \rangle$ – or, alternatively, $f(\mathsf{a}, \mathsf{b})$ –, where a is the subject, f is the predicate, and b is the object. Such triplets allow users to *extensionally* describe knowledge in a readable, machine-interpretable, and tractable way.

Collections of triplets constitute the so-called *knowledge graphs* (KG), i.e., directed graphs where vertices represent individuals, while arcs represent the binary properties connecting these individuals. These may explicitly or implicitly instantiate a particular *ontology*, i.e., a formal description of classes characterising a given domain, and of their relations (inclusion, exclusion, intersection, equivalence, etc.), as well as the properties they must (or must not) include.

Propositional logic. Propositional logic is a very restricted subset of FOL, where quantifiers, terms, and non-atomic predicates are missing. Hence, propositional formulæ simply consist of expressions involving one or many 0-ary predicates – i.e., *propositions* –, possibly interconnected by ordinary logic connectives. There, each proposition may be interpreted as a Boolean variable – which can either be true or false –, and the truth of formulæ can be computed as in the Boolean algebra. So, for instance, a notable example of propositional formula could be as follows: $p \wedge \neg q \rightarrow r$ where p may be the proposition "it is raining", q may be the proposition "there is a roof", whereas r may be the proposition "the floor is wet".

The expressiveness of propositional logic is far lower than the one of FOL. For instance, because of the lack of quantifiers, each relevant aspect/event should be explicitly modelled as a proposition. Furthermore, because of the lack of terms, entities from a given domain cannot be explicitly referenced. Such lack of expressiveness, however, implies computing the *satisfiability* of a propositional formula is a *decidable* problem—which may be a desirable property in some application scenarios.

Despite propositional logic may appear too trivial to handle common decision tasks where non-binary data is involved, it turns out a number of apparently complex situations can indeed be reduced to a propositional setting. This is the case for instance of any expression involving numeric variables or constants, arithmetical comparison operators, logic connectives, and nothing more than that. In fact, formulæ containing comparisons among variables constants (or among each others) can be reduced to propositional logic by mapping each comparison into a proposition.

2.3 eXplainable Artificial Intelligence (XAI)

Modern intelligent systems are increasingly adopting *sub-symbolic* predictive models to support their intelligent behaviour. These are commonly trained following a data-driven approach. Such wide adoption is unsurprising, given the unprecedented availability of data characterising the last decade. ML algorithms enable the detection of useful statistical information buried in data, semi-automatically. Information, in turn, supports decision-making, monitoring, planning, and forecasting virtually in any human activity where data is available.

However, despite its predictive capabilities, ML comes with some drawbacks making it perform poorly in critical use cases. The most relevant example is algorithmic *opacity*—intuitively, the human struggle to *understand* how ML-based systems operate or attain their decisions. In particular,

we refer to 'opacity' according to the third definition provided by Burrell [10]: "opacity as the way algorithms operate at the scale of application". In ML-based applications, complexity – and therefore opacity – arises because of the hardly predictable interplay among highly-dimensional datasets, the algorithms processing them, and the way such algorithms may change their behaviour during learning.

Opacity is a serious issue in all those contexts where human beings are liable for their decisions, or when they are expected/required to provide some sort of *explanation* for it—even if the decision has been suggested by software systems. This may be the case, for instance, in the healthcare, financial, or legal domains. In such contexts, ML is at the same time both an enabling factor – as it automates decision-making – and a limiting one—as opacity reduces human *control* on decision-making. The overall effect is general *distrust* w.r.t. AI-based solutions.

Opacity is also the reason why ML predictors are called 'black boxes' in the literature. The expression refers to systems where knowledge is not symbolically represented [32]. In absence of symbolic representations, *understanding* the operation of black boxes – or why they recommend or take particular decisions – becomes hard for humans. The inability to understand black-box content and operation may then prevent people from fully trusting (and, therefore, accepting) them.

To make the picture even more complex, current regulations such as the GDPR [53] are starting to recognise the citizens' *right to explanation* [21]—which eventually mandates *understandability* of intelligent systems. This step is essential to guarantee algorithmic fairness, to identify potential biases/problems in the training data or in the black box's operation, and to ensure that intelligent systems work as expected.

Unfortunately, the notion of understandability is neither standardised nor systematically assessed, yet. No consensus has been reached on what "providing an explanation" should mean when decisions are supported by ML [38]. However, many authors agree that black boxes are not equally *opaque*: some are more susceptible to interpretation than others for our minds—e.g., Figure 1 shows how differences in black-box interpretability are conventionally described.

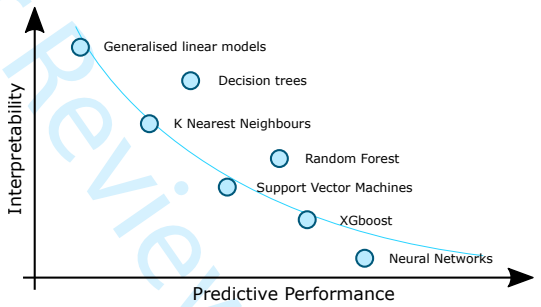


Fig. 1. Interpretability/performance trade-off for some common sorts of black-box predictors.

Despite being informal – as argued by [45], given the lack of measures for "interpretability" – Figure 1 effectively expresses why research on understandability is actually needed. Indeed, the figure stresses how the better performing black boxes are also the less interpretable ones. This is troublesome as, in practice, predictive performance can only rarely be preferred over interpretability.

Nevertheless, consensus has been reached about *interpretability* and *explainability* being desirable properties for intelligent systems. Hence, within the scope of this paper, we may briefly and informally describe XAI as the corpus of literature and methods aimed at making sub-symbolic AI more interpretable for humans, possibly by automating the production of explanations.

Along this line, based on the preliminary work by [15, 16], and by drawing inspiration from computational logic (and, in particular, model theory), we let 'interpretation' indicate *"the subjective relation that associates each representation with a specific meaning in the domain of the problem"*. In other words, interpretability refers to the cognitive effort required by human observers to assign a meaning to the way intelligent systems work, or motivate the outcomes they produce. In those contexts, the notion of interpretability is often coupled with properties as algorithmic transparency

(characterising approaches which are *not* opaque), decomposability, or simulatability—i.e., in a nutshell, *predictability*. Essentially, interpretable systems are understandable when humans can predict their behaviour.

As far as the term *explanation* is concerned, we trace back its meaning to the Aristotelian thought, other than the Oxford dictionary definition, which define 'explanation' as *"a set of statements or accounts that make something clear, or, alternatively, the reasons or justifications given for an action or belief"*. Thus, an explanation is an *activity* aimed at making the relevant details of an object clear or easy to understand to some observer.

Accordingly, the concepts of explainability and interpretability are basically *orthogonal*. However, they are not unrelated: explanations may consist of constructing better (≈ more-interpretable) representations for the black box at hand.

This is the case, for instance, of *explanation by model simplification* [48], where a poorly-interpretable model is translated into another – a more interpretable one –, having "high fidelity" [24] w.r.t. the first one. The translation process of the first model into the second one can be considered as an explanation. For example, as surveyed by this paper, several methods exist for *extracting* symbolic knowledge out of sub-symbolic predictors. When this is the case, the extraction act is technically an explanation, as it produces (more) interpretable objects – the symbolic knowledge – out of (less) interpretable ones—the predictors.

Conversely, one may regulate the interpretability of an opaque model by altering it to become "consistent" w.r.t. (i.e. "behave like") some more interpretable one. In this case, no explanation is involved, yet the resulting model has a higher degree of interpretability—which is commonly the goal. For instance, as discussed by this paper, several methods exist for *injecting* symbolic knowledge into sub-symbolic predictors. When this is the case, the injection act as the means by which opacity issues are worked around.

Interpretability and explainability are key enabling properties for making AI-based solutions (more) trustworthy in the eyes of human users. However, as highlighted by Rudin et al. [46], they are not necessarily sufficient: they may enable also *distrust*. In other words, interpretability and explainability enble finer control on intelligent systems, letting users decide whether to trust them or not. Along this line, the surveyed SKE/SKI methods should be regarded to as tools for increasing the degree of control users have on AI systems.

2.3.1 Sorts of explanation. According to the main impact surveys in the XAI area [6, 12, 24], two major approaches exist to bring explainability or interpretability features to intelligent systems, namely either *by-design* or *post-hoc*.

XAI by design. This approach to XAI aims at making intelligent systems interpretable or explainable *ex-ante*, since they are designed keeping these features as first-class goals. Method adhering to this approach can be further classified according to two sub-categories:

symbols as constraint containing methods supporting the creation of predictive models – possibly including or involving some black-box components – whose behaviour is constrained by a number of symbolic and intelligible rules, usually expressed in terms of (some subset of) first-order logic.

transparent box design containing methods supporting the creation of predictive models that are inherently interpretable, requiring no further manipulation;

In particular, in the remainder of this paper, we focus on methods from the latter category, as it is deeply entangled with symbolic knowledge *injection*.

Post-hoc explainability. This approach to XAI aims at making intelligent systems interpretable or explainable *ex-post*, i.e., by somehow manipulating poorly-interpretable pre-existing systems. Method adhering to this approach can be further classified according to the following sub-categories:

text explanation where explainability is achieved by generating textual explanations that help to explain the model results; methods that generate symbols representing the model behaviour are also included in this category, as symbols represent the logic of the algorithm through appropriate semantic mapping;

visual explanation techniques that allow the visualisation of the model behaviour; several techniques existing in the literature comes along with methods for dimensionality reduction, to make visualisation human-interpretable;

local explanation where explainability is achieved by first segmenting the solution space into less complex solution subspaces relevant for the whole model, then producing their explanation;

explanation by example allows for the extraction of representative examples that capture the internal relationships and correlations found by the model;

model simplification techniques where a completely-new simplified system is built, trying to optimise similarity with the previous one while reducing complexity;

feature relevance methods focus on how a model works internally by assigning a relevance score to each of its features, thus revealing their importance for the model in the output.

In particular, in the remainder of this paper, we focus on methods from the 'model simplification' category, as it is deeply entangled with symbolic knowledge *extraction*.

3 DEFINITIONS & METHODOLOGY

The goal of our SLR is to detect and categorise the many SKE and SKI algorithms proposed into the literature so far, hence shaping a clear picture of what SKE and SKI mean today.

Following this purpose, we *(i)* start from broad and intuitive definitions of both SKE and SKI (provided in Section 3.1); we then *(ii)* define a number of research questions aimed at delving into the details of actual SKE and SKI methods; along this line, we *(iii)* explore the literature looking for contributions matching the broad definitions from step *(i)* (following a strategy described in Section 3.2). Finally, by analysing such contributions, we *(iv)* provide answers for the research questions from step *(ii)* (in Section 4), and, in doing so, we *(v)* synthesise general, bottom-up taxonomies for both SKE and SKI (in Sections 4.1 and 4.2).

3.1 Definitions for Symbolic Knowledge Extraction and Injection

Here we provide broad definitions for both symbolic knowledge extraction and injection, following the purpose of drawing a line among what methods, algorithms, and technologies from the literature should be considered related to either SKE or SKI, and what should not. We do so under a XAI perspective, hence highlighting how both SKE and SKI help mitigating the opacity issues arising in data-driven AI. Then we discuss the potential arising from the *joint* exploitation of both SKE and SKI.

Notably, we tune our definitions so as to comprehend and generalise the many methods and algorithms surveyed later in this paper. Indeed, looking for a wider degree of generality, our definitions commit to no particular form of symbolic knowledge, nor sub-symbolic predictor—despite many surveyed techniques come with commitments of that sort. Hence, in what follows we write "symbolic knowledge" meaning "any chunk of intelligible information expressed in *any* possibly sort of logic", as well as any sort of information which can be rewritten in logic form (e.g., decision trees). Similarly, we write "sub-symbolic predictor" meaning "any sort of *supervised* ML model which can be fitted over *numeric* data to eagerly solve classification or regression tasks".

3.1.1 Extraction. Generally speaking, SKE serves the purpose of generating intelligible representations for the sub-symbolic knowledge a ML predictor has grasped from data during learning. Here we provide a general definition of SKE and discuss its purpose as well as the major benefits it brings against the XAI landscape.

Definition. We define SKE as

> *any* algorithmic *procedure accepting* trained *sub-symbolic predictors as input and producing* symbolic *knowledge as output, so that the extracted knowledge reflects the behaviour of the predictor with high* fidelity.

Notably, this definition emphasises a number of key aspects of SKE which are worth to be described in further detail.

First, SKE is modelled as a class of *algorithms* – hence finite-step recipes – characterised by what they accept as input and what they produce as output.

As far as the inputs of SKE procedures are concerned, the only explicit requirement is on *trained* ML predictors. There is no constraint w.r.t. the nature of the predictor itself, hence SKE procedures may be designed for any possible predictor family, in principle. Yet, this requirement implies that the predictor's training has already occurred, and it has reached some satisfying performance w.r.t. the task it has been trained for. Hence, in a ML workflow, SKE should occur *after* training and validation were concluded.

As far as the outputs of SKE procedures are concerned, the only explicit requirement is about the production of *symbolic* knowledge. "Symbolic" is here intended, in a broader sense, as a synonym of "intelligible" (for the human being), hence admissible outcomes are logic formulæ as well as decision trees, or bare human-readable text.

In any case, for an algorithm to be considered a valid SKE procedure, the output knowledge should mirror the behaviour of the original predictor w.r.t. the domain it was trained for, as much as possible. This involves some fidelity score aimed at measuring how well the extracted knowledge mimics the predictor it was extracted by, w.r.t. the domain and the task that predictor was trained for. This, in turn, implies that the extracted knowledge should, in principle, act as a predictor as well, thus being queryable as the original predictor would. Thus, for instance, if the original predictor is an image classifier, the extracted knowledge should let an intelligent agent classify images of the same sort, expecting the same result. The agent may then be either computational (i.e., a software program) or human, depending on whether the extracted knowledge is machine- or human-interpretable. Notably, the exploitation of *logic* knowledge as the target of SKE is of particular interest as it would enable both options.

Purpose and benefits. Generally speaking, one may be interested in performing SKE to inspect the inner operation of an opaque predictor, which should be considered a black box otherwise. However, one may also perform SKE to automatise and speed up the process of acquiring symbolic knowledge, instead of crafting knowledge bases manually.

Inspecting a black-box predictor through SKE, in turn, is an interesting capability within the scope of XAI. Given a black-box predictor and a knowledge-extraction procedure applicable to it, any extracted knowledge can be adopted as a basis to construct explanations for that particular predictor. Indeed, the extracted knowledge may act as an *interpretable replacement* (a.k.a. surrogate model) for the original predictor, provided that the two have a high fidelity score [15].

Accordingly, the application of SKE to XAI brings a number of relevant opportunities, e.g., by letting human users *(i)* study the internal operation of an opaque predictor to find, for instance, mispredicted input patterns; or correctly predicted input patterns leveraging upon some unethical decision process; *(ii)* highlight the differences or the common behaviours between two or more

black-box predictors performing the same task; *(iii)* merge the knowledge acquired by various predictors – possibly of different kinds – on the same domain—provided that the same representation format is used for extraction procedures [14].

3.1.2 Injection. Generally speaking, SKI serves a dual purpose w.r.t. to SKE. In particular, SKI aims at letting a ML predictor keep some symbolic knowledge into account when drawing predictions. Here, we provide a general definition of SKI, and we discuss its purpose and the major benefits it brings w.r.t. the XAI panorama.

Definition. We define SKI as

> *any* algorithmic *procedure affecting how sub-symbolic predictors draw their inferences in such a way that predictions are either* computed *as a function of, or made* consistent *with, some* given *symbolic knowledge.*

This definition emphasises a number of key aspects of SKI which are worth to be described in further detail. Similarly to SKE, it is modelled as a class of *algorithms*. Yet, dually w.r.t. extraction, SKI algorithms are procedures accepting symbolic knowledge as input and producing ML predictors as output.

About the inputs of SKI procedures, the only explicit requirement is that knowledge should be symbolic and user-provided—hence *human*-interpretable. However, since any input knowledge should be algorithmically manipulated by the SKI procedure, we elicit an implicit requirement here, constraining the input knowledge to be *machine*-interpretable as well. This implies that some formal language – e.g., some formal logic, or some decision tree – should be employed for knowledge representation, while free text or natural language should be avoided.

Along this line, another implicit requirement is that the input knowledge should be *functionally analogous* w.r.t. the predictors undergoing injection. In other words, if a predictor aims at classifying customer profiles as either worthy or unworthy for credit, then the symbolic knowledge should encode decision procedures to serve the exact same purpose, and observe the exact same information.

About the outcomes of SKI procedures, our definition identifies two relevant situations—which are not necessarily mutually-exclusive. On the one side, SKI procedures may enable sub-symbolic predictors to accept symbolic knowledge as input. SKI procedures of this sort essentially consist of pre-processing algorithm aimed at encoding symbolic knowledge in sub-symbolic form, hence enabling sub-symbolic predictors to accept them as input. In this sense, SKI procedures of this sort enable sub-symbolic predictors to (learn how to) *compute* predictions as functions of the symbolic knowledge they were fed with—assuming it has been conveniently converted into sub-symbolic form. On the other side, SKI procedures may alter sub-symbolic predictors so that they draw predictions which are *consistent* with the symbolic knowledge—according to some notion of *consistency*. SKI procedures of this sort essentially affect either the structure or the training process of the sub-symbolic predictors they are applied to, in such a way that the predictor must then keep the symbolic knowledge into account when drawing predictions. In this sense, SKI procedures of this sort force sub-symbolic predictors to learn not only from data but from symbolic knowledge as well.

In any case, regardless of their outcomes, SKI procedures fit the ML workflow in its early phases, as they may affect both preprocessing and training.

Notably, consistency plays a pivotal role in SKI, dually w.r.t. what fidelity does for SKE. Along this line, our definition involves some consistency score aimed at measuring how well the predictor undergoing injection can take advantage from the injected knowledge, w.r.t. the domain and the task that predictor was trained for. So, for instance, if a knowledge base states that loans

should be guaranteed to people from a given minority – as long as annual income overcomes a given threshold –, then any predictor undergoing injection of that knowledge base should output predictions respecting that statement—or at least minimise violations w.r.t. it.

Purpose and benefits. Generally speaking, one may be interested in performing SKI to reach a higher degree of control on what a sub-symbolic predictor is learning. In fact, SKI may either incentivise the predictor to learn some desirable behaviour, or discourage it from learning some undesired behaviour. However, one may also exploit SKI to perform sub-symbolic or fuzzy manipulations of symbolic knowledge, which would be otherwise unfeasible or hard to formalise via crisp symbols. While the latter option is further analysed by a number of authors – such as [1, 30] –, in the remainder of this section we focus on the former use case, as it is better suited to serve the purposes of XAI.

Within the scope of XAI, SKI is a remarkable capability as it provides a workaround for the issues arising from the opacity of ML predictors. While SKE aims at reducing the opacity of predictor by letting users observe its behaviour, SKI aims at bypassing the need for transparency. Indeed, predictors undergoing the injection of *trusted* symbolic knowledge provide higher guarantees about their behaviour, which will be more predictable and comprehensible—hence less astonishing.

Accordingly, the application of SKI to XAI brings a number of relevant opportunities, e.g., by letting the human designers *(i)* endow sub-symbolic predictors with their common sense, and, therefore: *(ii)* finely control what predictors are learning, and, in particular, *(iii)* let predictors learn about relevant situations, despite poor data is available to describe them. Provided that adequate SKI procedures exist, all such use cases come at the price of handcrafting *ad hoc* knowledge bases reifying the designers' common sense in symbols, and then injecting it in ordinary ML predictors.

3.2 Review Methodology

The overall review workflow is inspired by the goal question metric approach by [11]. In short, the workflow requires some clear research *goal(s)* to be fixed, and then decomposed into a number of research question the survey will then provide answers to. To produce such answers, the workflow requires of course scientific papers to be selected, and analysed. To serve this purpose, the workflow requires a pool of *queries* to be identified. Such queries must be performed on most relevant bibliographic search engines (e.g., Google Scholar, Scopus). Finally, the workflow requires the query results to be selected (or excluded) for further analyses following a reproducible criterion. Any subsequent analysis is then devoted to answer the aforementioned research questions, hence drawing useful classifications and general conclusions.

For the sake of reproducibility, in the remainder of this subsection we delve into the details of how our SLR on symbolic knowledge extraction and injection is conducted.

We start by defining three different research goals (**G**):

- **G1** – *"understanding which are the features of SKE algorithms"*,
- **G2** – *"understanding which are the features of SKI algorithms"*.
- **G3** – *"probing the current level of technological readiness of SKE/SKI technologies"*.

Then, we break them down in the following research questions (**RQ**):

- **RQ1** (from **G1**) – *"which sort of ML predictors can SKE be applied to?"*
- **RQ2** (from **G1**) – *"is there any requirement on the input data?"*
- **RQ3** (from **G1**) – *"which kind of SK can be extracted from ML predictors?"*
- **RQ4** (from **G1**) – *"for which kind of AI task can SKE be exploited?"*
- **RQ5** (from **G1**) – *"how does SKE work?"*
- **RQ6** (from **G2**) – *"which sorts of ML predictors can SKI be applied to?"*
- **RQ7** (from **G2**) – *"which kind of SK can be injected into ML predictors?"*

RQ8 (from **G2**) – *"for which kind of AI tasks can SKI be exploited?"*
RQ9 (from **G2**) – *"how does SKI work?"*
RQ10 (from **G3**) – *"which and how many SKE/SKI algorithms come with runnable software implementations?"*

Notice that research questions about SKE are analogous to those about SKI. In both cases, research questions are devoted to clarify which kind of information can SKE (resp. SKI) methods accept as input (resp. produce as output), how do they work, which AI tasks they can be used for (e.g., regression, classification), and which ML predictors they can be applied to (e.g., neural networks, SVM, etc.).

In order to answer the research questions above, we identify a number of queries to be performed on widely-available bibliographic search engines. In detail, queries involve the following keywords:

- ('rule extraction' ∨ 'knowledge extraction') ∧ ('neural networks' ∨ 'support vector machines')
- ('pedagogical' ∨ 'decompositional' ∨ 'eclectic') ∧ ('rule extraction' ∨ 'knowledge extraction')
- 'symbolic knowledge' ∧ ('deep learning' ∨ 'machine learning')
- 'embedding' ∧ ('knowledge graphs' ∨ 'logic rules' ∨ 'symbolic knowledge')
- 'neural' ∧ 'inductive logic programming'

As far as bibliographic search engines are concerned, we exploit Google Scholar[1], Scopus[2], Springer Link[3], ACM Digital Library[4], and DBLP[5].

For each search engine and query pair, we consider the first two pages of results. For each result, we inspect the title, abstract, and – in case of ambiguity –, the introduction, while trying and classifying it according to three disjoint circumstances: *(i)* the paper is a *primary work* describing some SKE or SKI method matching the broad definitions from 3.1, *(ii)* the paper is a *secondary work* surveying some portion of literature overlapping SKE or SKI (or both), *(iii)* the paper is *unrelated* w.r.t. to both SKE and SKI, hence it is not relevant for this survey. Notably, secondary works selected in step *(ii)* are valuable sources of primary works, hence we recursively explored their bibliographies to further select other primary works. In particular, in this phase we leverage upon relevant secondary works such as [4, 7, 12, 18, 24, 25, 27, 54, 55, 57, 61]—which we acknowledge as noteworthy (even though less extensive) surveys in the field of SKE or SKI.

We select 246 primary works, of which 129 works concern SKE, and 117 concern SKI. We then analyse each primary work individually, in order to provide answers to the aforementioned research questions. While doing so, we construct bottom-up taxonomies for both SKE and SKI.

Finally, we inspect each primary work for assessing its technological status. In particular, we look for runnable software implementations corresponding to the method described in the primary work. In case no software tool is clearly mentioned in the primary work, or if the software is not technically accessible (e.g., Web site or repository is private or non-reachable) at the time of the survey, then we consider the method as lacking software implementations. Otherwise, we further distinguish among methods coming with reusable software libraries, and methods coming with experimental code. In the first case, the software is ready for re-use, either because it is published on public software repositories such as PyPi, or because it is structured in such a way to let users exploit it for custom purposes. Vice versa, if the software tailored on the experiments mentioned in the primary work, then we consider it experimental.

[1]https://scholar.google.com
[2]https://www.scopus.com
[3]https://link.springer.com
[4]https://dl.acm.org
[5]https://dblp.uni-trier.de

4 SURVEY RESULTS

This section summarises the results of our survey. In particular, this is where we provide answers for the research questions outlined in Section 3.2.

Accordingly, we group research questions according to their main focus (SKE or SKI), and we answer to each question individually—grouping answers when convenient, for the sake of conciseness. Answers consist of brief statistical reports showing the distribution of the surveyed SKE/SKI methods w.r.t. some dimension of interest for either SKE or SKI. Interesting dimensions are presented on the fly, as part of our answers. This is deliberate, since we select as 'interesting dimension' any relevant way of clustering the surveyed methods. In other words, we let taxonomies emerge from the literature rather than super-imposing any particular view of ours.

4.1 Symbolic Knowledge Extraction

By building upon secondary works, such as the work by [12] and the survey of [4], we identify three relevant dimensions by which SKE methods can be categorised, namely: *(i)* the learning task(s) they support; *(ii)* the method's translucency; *(iii)* the shape of the extracted knowledge. By analysing the surveyed SKE methods, we find these categories adequate. However, we identify new dimensions, namely: *(iv)* the sort of input data the predictor undergoing extraction is trained upon, and *(v)* the expressiveness of the extracted knowledge. In what follows we answer research questions **RQ1**–**RQ5** by focusing on such dimensions, individually. Conversely, in the supplementary materials, we provide an overview of the 129 methods selected for SKE.

4.1.1 ***RQ1****: Which sort of ML predictors can SKE be applied to?* ***RQ5****: How does SKE work?* Answers for questions **RQ1** and **RQ5** are deeply entangled, as they are both related to SKE methods' *translucency*. Translucency deals with the need of SKE methods to inspect the internal structure of the underlying black-box model, while producing the extracted rules.

SKE methods provide for translucency in two ways [4], and can be labelled accordingly as

decompositional if the method needs to inspect (even partially) the internal parameters of the underlying black-box predictor, e.g., neuron biases or connection weights for neural networks, or support vectors for SVM;

pedagogical if the algorithm *does not* need to take into account any internal parameter, but it can extract symbolic knowledge by only relying on the predictor's outputs.

Along this line, we observe that surveyed SKE methods can be grouped into as many big clusters, depending on how they treat the predictor undergoing extraction.

W.r.t. **RQ1**, it is worth highlighting that pedagogical methods can be applied to any sort of supervised ML predictor, in principle—despite the literature may only report particular cases of application to specific predictors. Conversely, each decompositional method focuses on a specific sort of supervised ML predictor. Hence, decompositional SKE methods can be further categorised w.r.t. which sort of supervised ML predictors they are tailored upon. As detailed by Figure 2, the translucency is far from uniform for SKE methods. Indeed, nearly a half of the surveyed methods are pedagogical, while the rest are tailored on feed-forward neural networks (possibly, with fixed amounts of layers), SVM, linear classifiers, or decision tree ensembles.

W.r.t. **RQ5**, it is worth highlighting that pedagogical methods treat the underlying predictor as an *oracle*, to be queried for predictions the symbolic knowledge shall emulate. Conversely, decompositional methods must look into the internal structure of predictors, hoping to detect meaningful patterns in those patterns. For instance, SKE methods focusing on neural networks may try to interpret inner neurons as meaningful expressions combining their ingoing synapses.

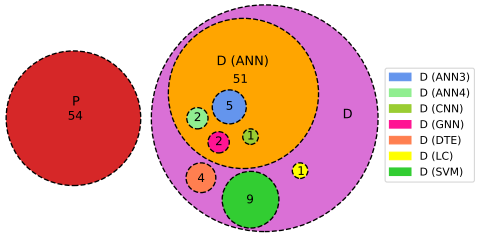


Fig. 2. Venn diagram categorising SKE methods w.r.t. *translucency*: pedagogical (P) or decompositional (D). For decompositional methods, we report the target predictor type: ANN⟨n⟩ = artificial neural networks (possibly, having exactly ⟨n⟩ layers), SVM = support vector machines, DTE = decision tree ensembles, LC = linear classifiers.

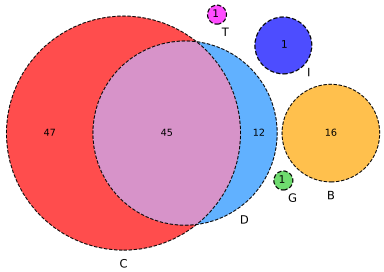


Fig. 3. Venn diagram categorising SKE methods w.r.t. the *input data type* required by the underlying predictor: binary (B), discrete (D), continuous (C), images (I), text (T), graphs (G).

4.1.2 ***RQ2****: Is there any requirement on the input data for SKE?.* This question can be answered by looking at the accepted input data type of the surveyed SKE methods. In most cases, data is structured, i.e. it consists of tables of numebrs, where features are of three different sorts:

binary if the feature can assume only two values, generally encoded with 0 and 1 (or -1 and 1, or `true` and `false`);

discrete if the feature can assume values drawn from a *finite* set of admissible values; notably, when this is the case, data science identifies two relevant sub-sorts of features: **ordinal** if the set of admissible values is *ordered* (hence, enabling the representation of the feature via some range of integer numbers), **categorical** if that set is *un*ordered (hence, enabling the representation of the feature via one-hot encoding);

continuous if the feature can assume any real numeric value.

Alternatively, data may consist of

images i.e. matrices of pixels, possibly with multiple channels;

text i.e. sequences of characters of arbitrary length;

graphs i.e. data structures of variable size, consisting of nodes/vertices interconnected by edges/arcs.

In Figure 3, we report absolute occurrence of the sorts of input features accepted by the surveyed SKE methods, as described by their authors. As the reader may notice, the vast majority of surveyed methods are tailored on structured data with *continuous* features.

4.1.3 ***RQ3****: Which kind of SK can be extracted from ML predictors?* Broadly speaking, any extracted SK should mirror (i.e., mimic) the operation of the ML predictor it has been extracted from. For *supervised* ML, this means the extracted knowledge should express a *function*, mapping input features into output features (e.g. classes, for classification tasks). Functions can be represented in symbols is several ways. Indeed, the SK extracted by the surveyed methods comes in various form.

Notably, such forms can be categorised under both a *syntactic* or *semantic* perspective. There, syntax refers to the *shape* of the extracted SK, whereas semantic refers to what kind of logic formalism the extracted knowledge may leverage upon—which is a matter of *expressiveness*.

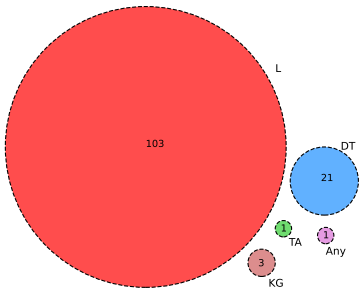


Fig. 4. Venn diagram categorising SKE methods w.r.t. the output knowledge's *shape*: rule lists (L), decision trees (DT) or tables (TA), knowledge graphs (KG).

Shape of the extracted knowledge. As far as syntax is concerned, decision *rules* [19, 28, 40] and *trees* [43, 44] are the most widespread human-comprehensible formats for the output knowledge, thus the vast majority of surveyed methods adopt one of these. However, other solutions have been exploited as well—e.g., decision *tables*. In all cases, however, a common trait is that functions of real numbers are expressed by using *symbols* to denote the same input and output features the underlying ML predictor was trained upon.

W.r.t. surveyed SKE methods, we identify four major admissible shapes:

lists of rules, i.e. sequences of logic rules to be read in some predefined order;
decision trees see Section 2.1.2;
decision tables i.e., concise visual rule representations specifying one or more conclusions for each set of different conditions. They can be *exhaustive* – if all the possible combinations are listed –, or *incomplete*—otherwise. Generally speaking, decision tables are structured as follows: there is a column (row) for each input and output variable and a row (column) for each rule. Each cell c_{ij} (c_{ji}) contains the value of the j-th variable for the i-th rule. An example of decision table is provided in the supplementary material.
knowledge graphs see Section 2.2.2.

Figure 4 sums up the occurrence of the different shapes of output rules required for SKE algorithms. As the reader may notice, the majority of the surveyed methods target rule *lists*. Arguably, this trend may be motivated by the great simplicity of rule lists, in terms of readability, and their algorithmic tractability.

Expressiveness of the extracted knowledge. Despite the extracted knowledge may contain statements of different shapes (e.g., rules, trees, tables), the readability, conciseness, and tractability of the extracted rules heavily depend on what can those statements contain—which, in turn, dictate what can (or cannot) be expressed. In the general case, statements may contain *predicates* or *relations* among the symbols representing input or output features. These may (or may not) contain logic connectives as well as arithmetic or logic comparators. SKE methods can be categorised w.r.t. which and how many ways of combining symbols are admissible within statements.

Along this line, we identify five major formats for statements in the surveyed SKE methods:

propositional rules are the simplest format, where statements consist of *propositions* – i.e. symbols denoting *boolean* input/output features –, possibly interconnected via logic connectives (negation, conjunction, disjunction, etc.). Notice that statements containing relations (e.g., arithmetic comparisons) among *single, continuous* features and *constant* values are indeed propositional as well.
fuzzy rules are propositional rules where the truth value of conditions and conclusions are not limited to 0 and 1, but can assume any value $\in [0, 1]$;

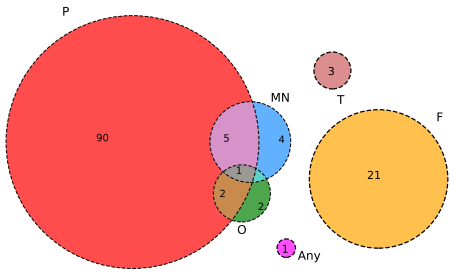


Fig. 5. Venn diagram categorising SKE methods w.r.t. the output knowledge's *expressiveness*: propositional (P), M-of-N (MN) , fuzzy (F), or oblique (O) rules; or triplets (T).

oblique rules have conditions expressed as inequalities involving linear combinations of the input variables. This is different from the propositional case, as features may be compared to other features (rather than constants alone).

***m*-of-*n* rules** are particular sorts of rules where *boolean* statements are grouped by n and each rule is *true* only if at least m literals (out of n) are *true*, with $m \leq n$. Notice that m-of-$(X_1, \ldots, X_n)$ is just a concise way of writing the disjunction among the conjunction of all possible m-sized combinations of n boolean literals $X_1, \ldots, X_n$. Hence, m-of-n rules are just a concise way of writing rules of other sorts: if $X_1, \ldots, X_n$ are all predicative statements, then supporting won't change their nature—and the same is true if $X_1, \ldots, X_n$ are oblique statements.

triplets see Section 2.2.2.

Figure 5 summarises the occurrence of the different SK formats produced by the surveyed SKE algorithms. As the reader may notice, the vast majority of surveyed SKE methods produce predicative rules, i.e. rules composed of several boolean statements about individual input features, possibly interconnected via logic connectives. Arguably, this trend may be motivated by the great tractability of propositional rules, and by their simplicity. In fact, to construct propositional rules, SKE algorithms may follow a *divide-et-impera* approach by focusing on each single input feature at a time—hence enabling the simplification of the extraction process itself.

4.1.4 ***RQ4****: For which kind of AI task can SKE be exploited?* ML methods are commonly exploited in AI to serve specific purposes, e.g. classification, regression, clustering, etc. Regardless of the particular means by which SKE is attained, extraction aids the human users willing to inspect *how* those methods work. However, the particular AI tasks ML predictors have been designed for play a pivotal role in determining what outputs users may expect from those predictors. A similar argument holds for extraction procedures, as the extracted knowledge should reflect the inner behaviour of the original predictor. Along this line, it is interesting to categorise SKE methods w.r.t. the AI task they assume for the ML predictors they are applied to.

Figure 6 summarises the occurrence of tasks among the surveyed SKE methods. Notably, most of them can be applied uniquely to *classifiers*, whereas a small portion of them is explicitly designed for *regressors*. Only few methods can handle both categories.

In general, we observe how the surveyed methods are tailored on either classification or regression tasks—when not both. In either cases, surveyed methods focus on supervised ML tasks. To the best of our knowledge, currently, there are no SKE procedures tailored on unsupervised or reinforcement learning tasks.

4.1.5 ***RQ10****: which and how many SKE algorithms come with runnable software implementations?* Among the 129 surveyed methods for SKE, we found runnable software implementations for 27 (20.9%). Of these, 10 consist of reusable software libraries, while the others are just experimental code. Figure 7 summurises this situation. In the supplementary materials, we provide details about

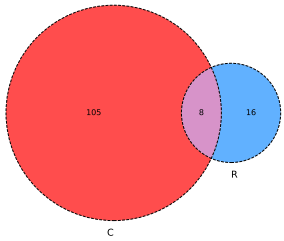


Fig. 6. Venn diagram categorising SKE methods w.r.t. the *targeted AI task*: classification (C) or regression (R).

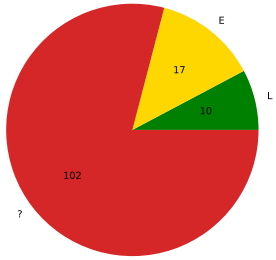


Fig. 7. Pie chart categorising SKE methods presence/lack of software implementations. There, 'L' denotes the presence of a reusable library, 'E' denotes experiments code, and '?' denotes lack of known technologies.

these implementations—there including the algorithm they implement and the link of the repository hosting the source code.

4.2 Symbolic Knowledge Injection

As far as SKI is concerned, we take into account no prior taxonomy. Indeed, despite the methods surveyed in this subsection come from well-studied (yet disjoint) research communities – such as neuro-symbolic computation [7] and knowledge graph embedding [55] –, we are not aware of any prior work attempting to unify these research areas under the SKI umbrella.

Along this line, we cluster the surveyed SKI methods according to four orthogonal dimensions, namely: *(i)* the type of SK they can inject, *(ii)* the strategy they follow to attain injection, *(iii)* the kind of predictors they can be applied to, *(iv)* the aim they pursue while performing injection. In what follows, we answer research questions **RQ6**–**RQ9** by focusing on such dimensions, individually. Conversely, in the supplementary materials, we overview the 117 methods selected for SKI.

4.2.1 ***RQ7****: Which kind of SK can be injected into ML predictors?* Generally speaking, SKI methods support the injection of knowledge expressed by various formalisms—despite each surveyed method focuses on some particular formalism. Along this line, a key discriminating factor is whether the chosen formalism is machine-interpretable or not—other than human-interpretable.

W.r.t. the formalism the input knowledge should adopt to support SKI, we may cluster the surveyed methods into two major groups, namely:

logic formulæ or **KB** (i.e., sets of formulæ) adhering to either FOL or some of its subsets, which are therefore both machine- and human-interpretable. Here, admissible sub-categories reflect the kinds of logics described in Section 2.2.1. Ordered by decreasing expressiveness, these are:

- **full first-order logic** formulæ including recursive terms, possibly containing variables, predicates of any arity, and logic connectives of any sorts, possibly expressing definitions;
- **Horn logic** (a.k.a. **Prolog**-like) where knowledge bases consist of head–body rules, involving predicates and terms of any sorts;

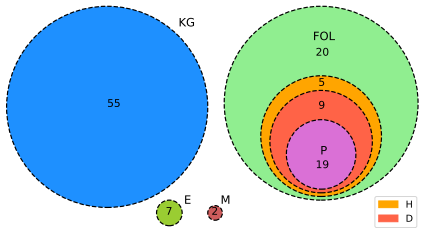


Fig. 8. Venn diagram categorising SKI methods w.r.t. the *input knowledge* type: knowledge graphs (KG), propositional logic (P), first-order logic (FOL), expert knowledge (E), Datalog (D), Horn logic (H), or modal logic (M).

Datalog i.e., Horn clauses without recursive terms (only constant or variable terms allowed);
modal logics i.e., extensions of some logic above with modal operators (e.g., $\Box$ and $\Diamond$), denoting the *modality* in which statements are true (e.g. *when*, in temporal logic);
knowledge graphs i.e., a particular application of description logics aimed at representing entity–relation graphs;
propositional logic where expressions are simply expressions involving boolean variables and logic connectives.

expert knowledge i.e., any piece of human- (but not necessarily machine-) interpretable knowledge by which data generation can be attained. This might be the case of physics formulae, syntactical knowledge, or any form of knowledge that is usually held by a set of human experts, and, as such, is only accessible to human beings. For this reason, expert knowledge injection requires some data to be generated to reify its information in tensorial form. Of course, expert knowledge may be cumbersome to extract and requires human engineers to take care of data generation before any injection can occur.

In Figure 8 we categorise the surveyed SKI methods w.r.t. their formalism of choice. There, KG are the most prominent cluster (including more than half of surveyed methods), while expert knowledge is the smallest one. Methods tailored on FOL or its subsets (apart from KG) form another relevant cluster. There, propositional logic plays a pivotal role, as it involves the relative majority of methods.

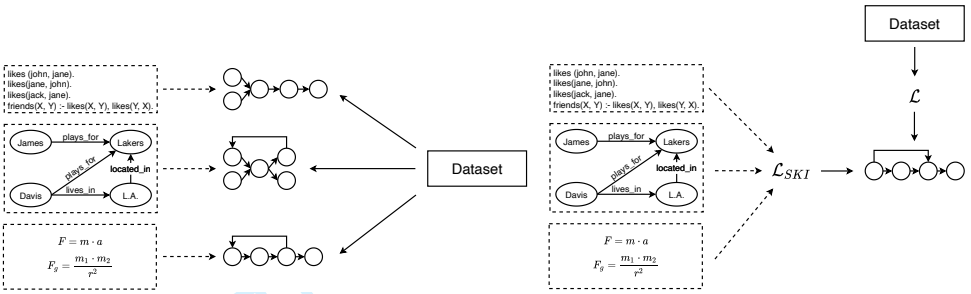
Notably, as long as the logic formalism is concerned, we consider and report the *actual* logic used in the papers. Indeed, this is rarely explicitly stated by the authors into their papers. So, we deduce the actual logic used by each SKI method from the constraints its logic is subject to, according to its authors.

4.2.2 ***RQ9****: How does SKI work?* By analysing the surveyed SKI methods, we acknowledge great variety in the actual way injection is performed. Arguably, however, such variety can be tackled by focusing on tree major *strategies*, depicted in Figure 9 and summarised below:

predictor structuring where (a part of) a sub-symbolic predictor (commonly, NN) is created to mirror the symbolic knowledge via its own internal structure. In other words, a predictor is created or extended to mimic the behaviour of the SK to be injected. For example, when it comes to NN, their internal structure is crafted to represent logic predicates via neurons, or and logic connectives via synapses;

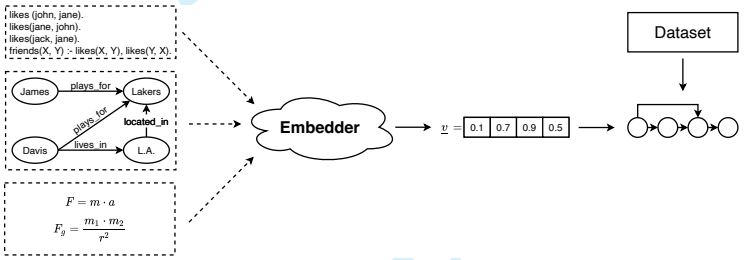
knowledge embedding where SK is converted into numeric-array form – e.g., vectors, matrices, tensors, etc. – to be provided as "ordinary" input for the sub-symbolic predictor undergoing injection. In other words, numeric data is generated out of symbolic knowledge. Any numeric representation of this sort is called *embedding* [of the original symbolic knowledge]. For example, this is the common strategy exploited by the knowledge graph embedding community [55], as well as by graph NN [1, 30];

Fig. 9. Overview of major strategies followed by surveyed SKI methods.



(a) Structuring strategy: a (portion of a) neural network is constructed, mirroring the symbolic knowledge.

(b) Guided learning strategy.



(c) Embedding strategy: the symbolic knowledge is converted in tensorial form and ML predictors are fed "as usual".

guided learning (a.k.a., **constraining**) where SK is used to steer the learning process of ML predictors, by either penalising inconsistent behaviours or by incentivising consistent behaviours w.r.t. the SK. When the predictor undergoing injection is trained via some optimisation process involving loss functions being minimised (e.g., NN), guided learning is achieved by altering those loss functions in such a way that violations w.r.t. the SK increase the loss. A dual statement holds for predictors requiring training to step through maximization processes. The recent book by [22] nicely overviews methods of these kinds.

Figure 10 summarises the frequency of these strategies among the surveyed SKI algorithms. Notably, the distribution of surveyed SKI methods among the three categories above is balanced.

4.2.3 ***RQ6****: Which sorts of ML predictors can SKI be applied to?* Virtually *all* surveyed SKI methods are designed to inject knowledge into neural networks. However, as this survey spans over 2 decades, the sorts of NN supported by SKI methods are manifold—despite each method is tailored on specific sorts of NN.

Accordingly, surveyed SKI methods can be classified w.r.t. the particular sort of NN they support. As detailed by Figure 11, admissible choices along this line fit the many sorts of NN discussed in Section 2.1.2, namely:

feed-forward NN multi-layered NN where neurons from layer i are only connected with layer $i + 1$, and multiple (≥ 2) layers may exist;

convolutional NN particular cases of feed-forward NN, involving convolutional layers as well;

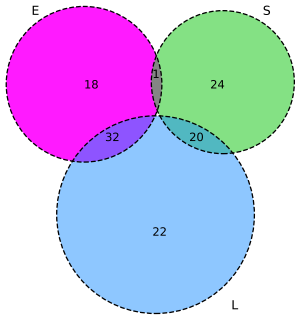


Fig. 10. Venn diagram categorising SKI methods w.r.t. *strategy*: structuring (S), embedding (E), or guided learning (L).

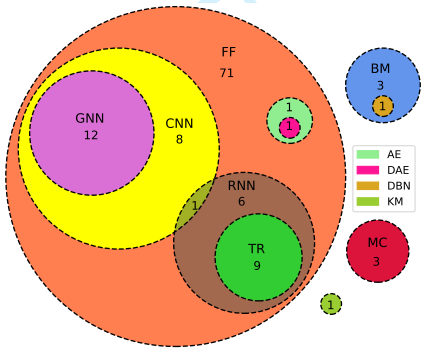


Fig. 11. Venn diagram categorising SKI methods w.r.t. the *targeted predictor* type: feed-forward (FF), convolutional (CNN), graph (GNN) or recurrent (RNN) neural networks, Boltzmann machines (BM), Markov chains (MC), transformers (TR), auto-encoders (AE), deep belief networks (DBN), denoising auto-encoders (DAE), kernel machines (KM).

graph NN particular cases of convolutional NN tailored on graph-like data;
recurrent NN particular cases of NN admitting loops among layers;
Boltzmann machine a particular neural architecture where connections are undirected—i.e., every node is connected to every other node;
transformer particular case of recurrent NN that leverage a self-attention mechanism—i.e., differentially weighting parts of the input data depending on their significance;
auto-encoders particular case of feed-forward NN, characterised by a bottleneck architecture used to learn reduced data encodings through learning to regenerate the input from the encoding;
deep belief networks a composition of multiple Boltzmann machines, stacked altogether, in a feed-forward fashion;
denoising auto-encoder particular case of auto-encoders working over corrupted input.

Notable exceptions are:

kernel machines ML models relying on kernels—i.e., similarity measures between observed patterns;
Markov chains state machines with probabilities on state transitions, modelling stochastic phenomena.

Interesting enough, the vast majority of methods rely on (some sort of) NN. The reason is straightforward: methods tailored upon GNN (resp. CNN) assume the networks to accept specific kinds of data as input, e.g. graphs (resp. images), while ordinary feed-forward NN accept raw vectors of real numbers.

4.2.4 ***RQ8****: For which kind of AI tasks can SKI be exploited?* Unlike SKE methods – which uniquely serve the purpose of inspecting black-box predictors by mimicking the way they address supervised

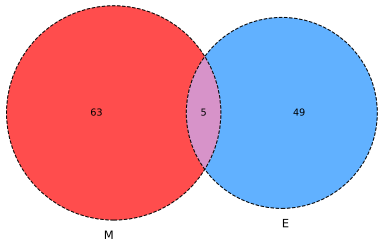


Fig. 12. Venn diagram categorising SKI methods w.r.t. *aim*: knowledge manipulation (M) or enrich (E).

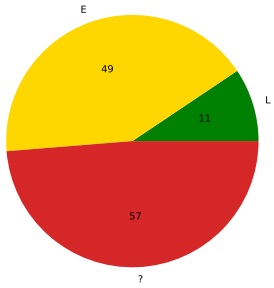


Fig. 13. Pie chart categorising SKI methods presence/lack of software implementations. There, 'L' denotes the presence of a reusable library, 'E' denotes experiments code, and '?' denotes lack of known technologies.

learning tasks –, SKI methods from the literature may serve multiple purposes. As outlined by Figure 12, we identify two major purposes SKI methods may pursue, by targeting either symbolic or sub-symbolic AI tasks. More precisely, SKI methods may pursue:

symbolic knowledge manipulation where SKI enables the *sub*-symbolic manipulation of *symbolic* knowledge, by letting sub-symbolic predictors treat SK similarly to what done by symbolic engines. In doing so, SKI supports symbolic-AI tasks such as

- **logic inference** in its many forms (e.g. deductive, inductive, probabilistic, etc.), i.e. drawing conclusions out of symbolic KB;
- **information retrieval** looking for information in symbolic KB;
- **KB completion** finding (and adding) missing information in symbolic KB;
- **KB fusion** merging several KB into a single one, take care of (possibly, syntactically different) overlaps;

The key point here is supporting tasks where both inputs and outputs are symbolic in nature, but leveraging upon sub-symbolic methods to gain speed, fuzziness, and robustness against noise.

learning support (a.k.a., **enrich**) where SKI lets *sub*-symbolic methods consume *symbolic* knowledge to either improve or enrich learning capabilities. In doing so, SKI supports ordinary ML tasks – such as classification –, by allowing ML predictors to process (or take advantage by) structured symbolic knowledge. The underlying idea of such approaches is that there exist some concepts that are cumbersome or troublesome to learn from examples—e.g., syntactical concepts, semantics, etc. Therefore, symbolic knowledge expressing these high-level concepts may be injected directly into the model to be trained.

As the reader may note from the picture, surveyed SKI methods are quite balanced w.r.t. the categories above.

4.2.5 ***RQ10**: which and how many SKI algorithms come with runnable software implementations?*

Among the 117 surveyed methods for SKI, we found runnable software implementations for 60

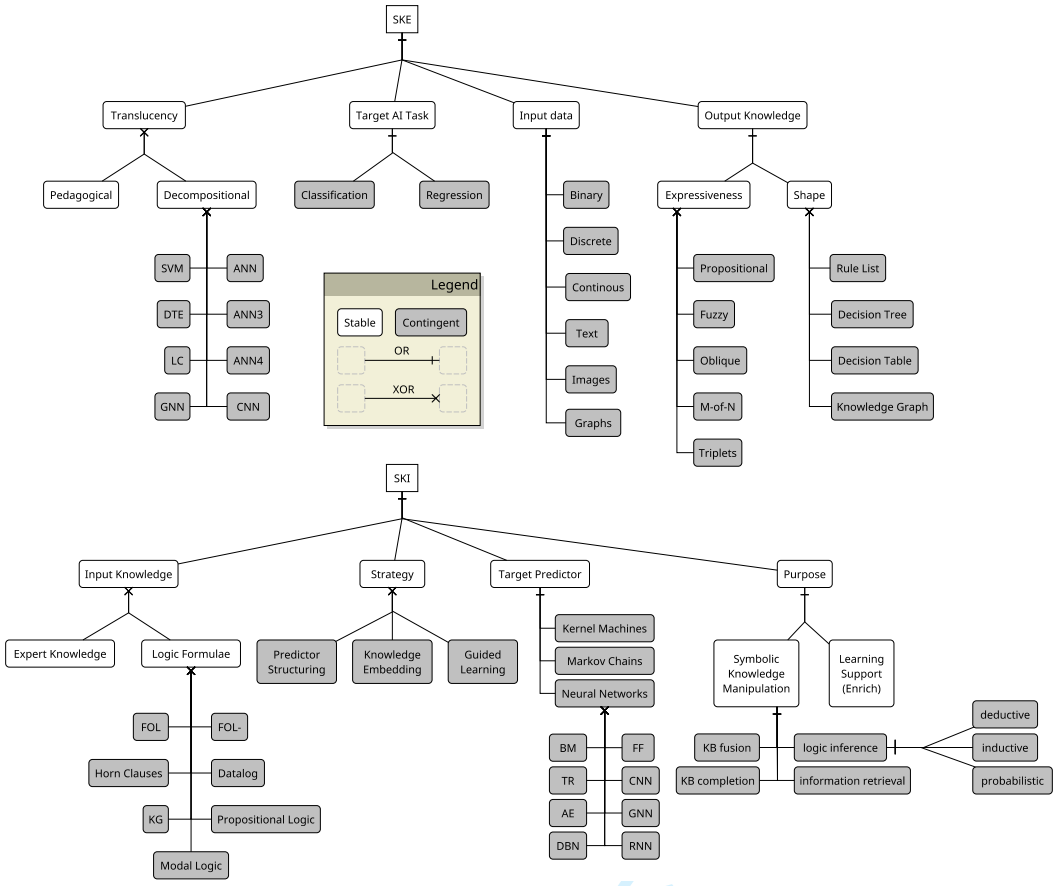


Fig. 14. Summary of SKE and SKI taxonomies derived from the literature, as discussed in section 4.

(51.3%). Of these, 11 consist of reusable software libraries, while the others are just experimental code. Figure 13 summurises this situation. In the supplementary materials, we provide details about these implementations—there including the algorithm they implement and the link of the repository hosting the source code.

5 DISCUSSION

Figure 14 summarises the main contribution of our paper—i.e., the taxonomies for SKE and SKI we induced from the surveyed literature. Generally speaking, such taxonomies are useful tools to categorise present (and, hopefully, future) SKE/SKI methods, and to highlight the relevant features of each particular method. In this way, the interested readers may figure out what to expect from any given SKE/SKI method, as well as draw general analyses concerning the state of the art. Accordingly, in this section we analyse our taxonomies, elaborating on the current challenges and future perspectives.

It is worth mentioning that our taxonomies involve both “stable” and “contingent” categories by which SKE/SKI methods can be described. These are represented as either white or grey boxes in Figure 14. Stable categories are time-independent and they are not susceptible to change in the near future, while contingent categories are subject to trends and may evolve. Consider for instance

SKE methods (see Figure 14), categorised w.r.t. their output knowledge. While expressiveness is a stable sub-category, its actual sub-sub-categories are contingent, meaning that new ones may be added in the future.

5.1 SKE Taxonomy

As shown in Figure 14, SKE methods can be classified by *(i)* translucency, *(ii)* targeted AI task, *(iii)* nature of the input data, and *(iv)* form of the output knowledge. W.r.t. item *(i)*, SKE methods can either be categorised as pedagogical or decompositional. In the particular case of decompositional methods, the actually targeted predictor is relevant too—and possibilities *currently* include NN, decision trees, SVM, and linear classifiers. W.r.t. item *(ii)*, SKE methods may target classification or regression tasks, or both. In any case, they *currently* target supervised ML tasks alone. W.r.t. item *(iii)*, SKE methods accept predictors trained upon binary, discrete, or continuous data. Finally, w.r.t. item *(iv)*, SKE methods may produce symbolic knowledge of different shapes, and with different expressiveness. Shapes may *currently* involve rule lists, as well as decision trees or tables. Conversely, as long as expressiveness is involved, symbolic knowledge may be propositional, oblique, or fuzzy—possibly including M-of-N-like statements.

About translucency. It is worth stressing the relevance of pedagogical methods from the engineering perspective. Indeed, if properly implemented, pedagogical methods may be exploited in combination with predictors of any sorts. Of course, they are expected to reach lower performances w.r.t. decompositional ones, as they access less information. On the other side, decompositional methods may be more precise at the expense of generality.

About input data. We recall that binary features are particular cases of discrete features, while discrete features are, in turn, particular cases of continuous features. Hence, it is worthwhile noticing that extractors requiring only binary features can be applied to categorical datasets by pre-processing discrete attributes via one-hot encoding (OHE). Analogously, extractors requiring discrete features can work with continuous attributes if those continuous features are *discretised*. Finally, continuous features can be converted into binary ones by performing discretisation and OHE, in this exact order.

While these transformations can always be applied in the general case, some authors have included them in their SKE methods at the design level. Hence, some papers explicitly account discretisation or OHE as part of the SKE methods they propose. This is the case, for instance, of the methods labelled as "C+D" in Figure 3. Other methods may instead rely upon other discretisation strategies, such as the ones surveyed by [59]

About output knowledge. It is worth stressing that differences among rule lists, decision trees, and tables are mostly syntactic, as conversions among these forms are possible in the general case (cf. the supplementary materials for examples). As far as expressiveness is concerned, we remark that all logic formalisms currently in use for SKE are essentially particular cases of propositional logic—possibly, under a fuzzy interpretation. This implies that the full power of FOL is far from being fully exploited in practice.

Finally, we point out some correlations among the expressiveness of output rules and the nature of the predictor they are extracted from, as well as the input data it is trained upon. For instance, *decompositional* SKE methods focusing upon NN are more likely to adopt M-of-N statements. Arguably, the reason is that M-of-N expressions aggregate several elementary statements into a single formula, similarly to how neurons aggregate synapses from previous layers in NN—hence such methods approximate neurons via M-of-N expressions. Another example: SKE methods working with *continuous* input data are more likely to adopt oblique rules—or, at least, propositional

rules with arithmetic comparisons. In fact, decisions are there drawn by comparing numeric variables with constants or among each other.

On SKE methods' chronology. In conclusion, we stress the chronological distribution of SKE methods. As highlighted by the supplementary materials, the majority of SKE methods have been proposed during the decades ranging from the 90s up to the 2010s. Contributions slowed down after that, up to the 2020s, where SKE gained new momentum.

In our opinion, research on ML interpretability gained momentum more than once in the history of AI. Each time sub-symbolic AI attracted the interest of researchers, so did the need to make it more comprehensible. Arguably, this is the reason why most SKE-related works are concentrated around the 2000s. In the last years, we are witnessing the novel spring of sub-symbolic AI [35], which is, in turn, motivating researchers' interest in XAI. Arguably, this is why SKE is gaining novel momentum in recent years.

5.2 SKI Taxonomy

As shown in Figure 14, SKI methods can be classified by *(i)* form of the input knowledge, *(ii)* followed strategy, *(iii)* targeted predictor type, and *(iv)* purpose. W.r.t. Item *(i)*, SKI methods can either accept logic formulæ or expert knowledge as input. In the former case, *current* possibilities include FOL and its subsets, and in particular knowledge graphs. W.r.t. Item *(ii)*, SKI methods may *currently* follow one of three strategies, namely: predictor structuring, knowledge embedding, or guided learning. W.r.t. Item *(iii)*, SKI methods *currently* mostly target NN-based predictors, other than Markov chains and kernel machines. Finally, w.r.t. Item *(iv)*, SKI methods may pursue two kinds of purposes, non-exclusively: manipulating symbolic knowledge or supporting/enriching learning. In the former case, *current* possibilities involve symbolic AI related tasks such as logic inference (and its many forms), information retrieval, and KB completion/fusion.

About input knowledge and injection strategies. Logic formulæ are the most common approach to define prior concepts to be injected. This is true, in particular, for SKI approaches following the model structuring or guided learning strategies. Indeed, via logic formulæ, they express criteria that sub-symbolic models should satisfy or emulate. However, methods of these sorts often require formulæ to be grounded. Grounding introduces computational burden and hinders capability of representing recursive or infinite data structures—hence limiting what can actually be injected.

Conversely, knowledge graphs are the most common knowledge representation approach when it comes to perform SKI following the knowledge embedding strategy. This is unsurprising, given that "knowledge graph embedding" is a research line *per se*.

About target predictors. Neural networks play a pivotal role in SKI. Arguably, the reason lays in the great *malleability* of NN w.r.t. their structure and training, as well as their *flexibility* w.r.t. feature learning. In fact, NN come in different shapes as different architectures may be constructed by connecting neurons in various ways. This is fundamental to support SKI via *predictor structuring*. Furthermore, as long as their architectures are DAG, NN can be trained via gradient descent, i.e. by minimising a loss function of arbitrary shape. This is, in turn, fundamental to support SKI via *guided learning*. Finally, feature learning is a characterising capability of NN, making them capable to automatically elicit the relevant aspects they should focus upon, w.r.t. input data. This is the reason why NN are well suited for the knowledge embedding strategy as well. Accordingly, to the best of our knowledge, there exists no other sort of predictor having similar flexibility and malleability.

On SKI methods' chronology. In conclusion, we stress the chronological distribution of SKI methods. As highlighted by the supplementary materials, the majority of SKI methods have been proposed after 2010, and, notably, the amount of contribution has exploded after 2015.

In our opinion, this distribution is due to the composite effect of three major drivers, namely: natural language processing (NLP), XAI, and neuro-symbolic computation (NSC). Arguably, all such drivers are gaining momentum in the last years, due to the success of machine- and deep-learning. Indeed, NLP reached unprecedented performance levels after it started leveraging DL, possibly combined with knowledge graphs and the corresponding SKI methods. Similarly, a portion of XAI-related contributions propose SKI methods aimed at controlling, constraining, or guiding what predictors learn from data. Finally, NSC has recently emerged as a field exploiting SKI methods to process logic knowledge sub-symbolically, by exploiting the malleability of NN.

5.3 Challenges

We observe that SKE algorithms focus exclusively on supervised learning tasks – i.e., classification and regression – while they do not tackle unsupervised or reinforcement learning tasks—e.g., clustering or optimal policy search. One may argue, for instance, that clustering algorithms are not opaque – e.g., K-nearest neighbours –, despite operating on numeric data. However, *pedagogical* SKE algorithms could be used on clustering predictors with no or minimal adjustments—as trained clustering predictors are essentially classifiers upon anonymous classes. Similarly, it could be possible to perform extraction on predictors trained using reinforcement learning with existing SKE algorithms. Indeed, future literature works on SKE for unsupervised learning would be needed.

Furthermore, the vast majority of SKI algorithms accept knowledge in form of knowledge graph – a.k.a., description logic – or propositional logic (Figure 8), which are much less expressive than FOL. These logics lack support for recursion and function symbols, meaning that the user is quite limited in providing knowledge to predictors. The reason behind this is that common ML predictors are acyclic (e.g., NN, etc.), meaning that there is no straightforward way to integrate recursion nor indefinitely deep data structures without severe information loss due to approximations. Hence, future research efforts concerning SKI may consider addressing the injection of logics involving recursive clauses or arbitrarily deep data structures.

5.4 Opportunities

We propose a brief discussion on the benefits arising from the *joint* exploitation of both SKI and SKE in the engineering of AI solutions. *(i)* the possibility of *debugging* sub-symbolic predictors, and *(ii)* the exploitation of symbolic knowledge as the *lingua franca* among heterogeneous hybrid systems. Accordingly, in the remainder of this sub-section, we delve into the details of these expected benefits.

5.4.1 Debugging sub-symbolic predictors. Debugging is a common activity for software programmers: it aims at spotting and fixing *bugs* in computer programs under production/maintenance. There, a bug is some unknown error contained into the program which leads to some unexpected or undesired observable behaviour of the computer(s) running that program. The whole procedure relies on the underlying assumption that computer programs are intelligible to the programmer debugging them, and that the program can be precisely edited to fix the bug.

One may consider XAI techniques as means to "debug" sub-symbolic predictors. In this metaphor, sub-symbolic predictors correspond to computer programs – despite they are not manually written by programmers, but learned from data –, while data scientists correspond to programmers. However, debugging sub-symbolic predictors is hard, because of their opacity – which makes their inner behaviour poorly intelligible for data scientists –, and because they cannot be precisely edited

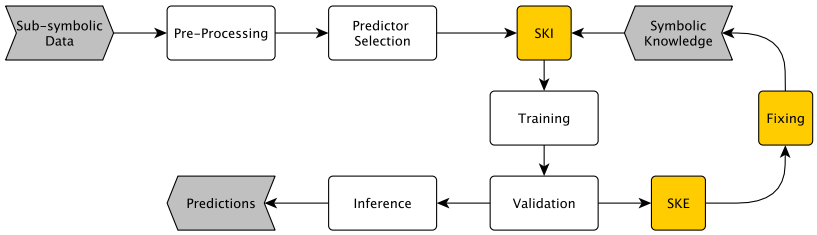


Fig. 15. ML workflow enriched with SKI and SKE phases. On the right, the train-extract-fix-inject loop is represented.

after training—but should be rather retrained from scratch. Accordingly, here we discuss the role of SKE and SKI to let data-science in overcoming these issues, hence allowing data scientists to debug sub-symbolic predictors.

Figure 15 provides an overview of how SKI and SKE fit the generic ML workflow. In particular, the figure stresses the relative position of both SKI and SKE w.r.t. the other phases of the ML workflow. Notably, SKI should occur before (or during) training, while SKE should occur after it. However, the figure also stresses the addition of a *loop* into an otherwise linear workflow (right-hand side of the figure). We call it the "train-extract-fix-inject" (TEFI) loop, and we argue it is a possible way to debug sub-symbolic predictors.

In the TEFI loop, SKE is the basic mechanism by which the inner operation of a sub-symbolic predictor (i.e., 'the program', in the metaphor) is made intelligible to data scientists. The extracted knowledge may then be understood by data scientists and "debugged"—hence looking for pieces of knowledge which are wrong w.r.t. the data scientists expect. Then, symbolic knowledge may be precisely edited and fixed. Along this line, SKI is the basic mechanism by which a trained predictor is precisely edited to adhere to the fixed symbolic knowledge.

5.4.2 Symbolic knowledge as the lingua franca *for intelligent systems.* Intelligent systems can be suitably modelled and described as composed of several intelligent, heterogeneous, and hybrid computational agents interoperating – and possibly communicating – among each others. There, a computational agent is any software or robotic entity capable of computing, other than perceiving and affecting some given environment—be it the Web, the physical world, or anything in between. To make the overall systems intelligent, these agents should be capable of a number of intelligent behaviours, ranging from image, speech, or text recognition to autonomous decision making, planning, or deliberation. Behind the scenes, these agents may (also) leverage upon sub-symbolic predictors – possibly trained upon locally-available data –, as well as symbolic reasoners, solvers, or planners to support such sorts of intelligent behaviours. In this sense, such agents are *hybrid*, meaning that they involve both symbolic and sub-symbolic AI facilities. However, interoperability may easily become a mirage because of *(i)* the wide variety of algorithms, libraries, and platforms supporting sub-symbolic ML, other than *(ii)* the possibly different data items each agent may locally collect and later train predictors upon. Indeed, each agent may learn (slightly) different behaviours, due to the differences in the training data and in the actual ML workflow it locally adopts. When this is the case, exchange of behavioural knowledge may become cumbersome or infeasible.

In such scenarios, SKI and SKE may be enablers of a higher degree of interoperability, by supporting the exploitation of symbolic knowledge as the *lingua franca* for heterogeneous agents. Indeed, hybrid agents may exploit SKE to extract symbolic knowledge out of their local sub-symbolic predictors, and exchange (and possibly improve) that symbolic knowledge with other

agents. Then, any possible improvement of the symbolic knowledge attained via interaction may be back-propagated into local sub-symbolic predictors via SKI, hence enabling agents' behaviour to improve as well.

5.5 Limitations

This SLR also means to be as comprehensive, precise, and reproducible as possible; nonetheless, we acknowledge two potential limitations: *(i)* the expected life span of our taxonomies, and *(ii)* terminology issues in the literature.

Both SKE and SKI are hot topics nowadays: further advancements have to be expected for the next decade, at least. Hence, our taxonomies may require to be verified, and possibly updated, sometime in the future. The straightforward methodological approach defined by our SLR, however, should ensure a clear path to future reproductions of this work.

Also, an evolution in the naming conventions clearly emerge from our analysis. Along the years, SKE has been called in disparate ways—e.g. "rule extraction" [4], or "knowledge distillation" [58], just to name a few. The same holds for SKI: there, naming conventions are commonly based on the injection strategy, yet they rarely contain the word "injection". So, we may have missed some works while collecting papers simply because they were using different naming conventions that we were not able to devise out. This is an inherent issue of the keyword-based methodology we adopted for SLR. To minimise issues in the classifications of present and future SKE/SKI methods, we draw loose definitions, and carefully read papers to determine whether they match our definitions or not. Yet, missing works for unexpected terminology choices cannot be excluded.

6 CONCLUSION

In this paper we survey the state of the art of symbolic knowledge extraction and injection under a XAI perspective. Stemming from two original definitions, we *systematically* explore the literature of both SKE and SKI, spanning a period of three decades. Our goal is to elicit the major characteristics of SKE/SKI algorithms form the literature (**G1**–**G2**), hence deriving general taxonomies which we hope other reserchers may exploit. Another goal of ours is to assess the current state of technologies (**G3**) by indentifying software implementations of SKE/SKI techniques.

Considerable efforts were spent in keeping our review *reproducible*—as prescribed by the goal question metric approach by [11]. Along this line, we design ten research questions (**RQ1**–**RQ10**), and we engineer *ad hoc* queries to be performed on most relevant search engines for scientific literature. We select 246 primary works, almost evenly distributed among SKE and SKI, other than 11 secondary works. By analysing these papers, we define and discuss two general taxonomies for both SKE and SKI, which are general enough to categorise present (and possibly future) methods.

Roughly, surveyed methods are categorised w.r.t. what they accept as input and produce as output (in terms of symbolic knowledge or predictors), other than how they operate and why. We also collect data about which and how many SKE/SKI methods come with runnable software implementations (namely, 87, i.e. 35.4%). In the supplementary materials, we also report Web homepages for the available implementations.

Overall, the implications of our study are manifold. First, our SRL demonstrates how SKE and SKI are hot topics of AI research, nowadays. The literature already contains hundreds of contributions and our taxonomies provide a compact, yet general, criterion for navigating it. Hopefully, our SLR will serve as a map for future contributions, which we expect to flourish soon and abundantly.

ACKNOWLEDGMENTS

This paper has been partially supported by *(i)* the CHIST-ERA IV project "EXPECTATION" (CHIST-ERA-19-XAI-005), co-funded by European Union (EU) and the Italian MUR (Ministry for University

and Research); and by *(ii)* the EU's Horizon Europe research and innovation programme (G.A. 101070363).

REFERENCES

[1] Andrea Agiollo, Giovanni Ciatto, and Andrea Omicini. 2021. Graph Neural Networks as the Copula Mundi between Logic and Machine Learning: A Roadmap. In *WOA 2021 – 22nd Workshop "From Objects to Agents"* (Bologna, Italy) *(CEUR Workshop Proceedings, Vol. 2963)*, Roberta Calegari, Giovanni Ciatto, Enrico Denti, Andrea Omicini, and Giovanni Sartor (Eds.). CEUR-WS.org, Bologna, Italy, 98–115. http://ceur-ws.org/Vol-2963/paper18.pdf 22nd Workshop "From Objects to Agents" (WOA 2021), Bologna, Italy, 1–3 Sept. 2021. Proceedings.

[2] Andrea Agiollo, Giovanni Ciatto, and Andrea Omicini. 2021. *Shallow2Deep*: Restraining Neural Networks Opacity through Neural Architecture Search. In *Explainable and Transparent AI and Multi-Agent Systems. Third International Workshop, EXTRAAMAS 2021*, Davide Calvaresi, Amro Najjar, Michael Winikoff, and Kary Främling (Eds.). Lecture Notes in Computer Science, Vol. 12688. Springer, 63–82. https://doi.org/10.1007/978-3-030-82017-6_5

[3] Miklós Ajtai and Yuri Gurevich. 1994. Datalog vs First-Order Logic. *J. Comput. Syst. Sci.* 49, 3 (1994), 562–588. https://doi.org/10.1016/S0022-0000(05)80071-6

[4] Robert Andrews, Joachim Diederich, and Alan B. Tickle. 1995. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems* 8, 6 (1995), 373–389. https://doi.org/10.1016/0950-7051(96)81920-4

[5] Franz Baader. 2003. Basic description logics. In *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, USA, 43–95.

[6] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. 2020. Explainable Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58, December 2019 (2020), 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

[7] Tarek R. Besold, Artur S. d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro M. Domingos, Pascal Hitzler, Kai-Uwe Kühnberger, Luís C. Lamb, Daniel Lowd, Priscila Machado Vieira Lima, Leo de Penning, Gadi Pinkas, Hoifung Poon, and Gerson Zaverucha. 2017. Neural-Symbolic Learning and Reasoning: A Survey and Interpretation. *Neuro-Symbolic Artificial Intelligence: The State of the Art* 342 (2017), 1–51. https://doi.org/10.3233/FAIA210348 arXiv:1711.03902

[8] Ronald J. Brachman and Hector J. Levesque. 2004. The Tradeoff between Expressiveness and Tractability. In *Knowledge Representation and Reasoning*, Ronald J. Brachman and Hector J. Levesque (Eds.). Morgan Kaufmann, San Francisco, 327–348. https://doi.org/10.1016/B978-155860932-7/50101-1

[9] Leo Breiman, Jerome Friedman, Charles J. Stone, and Richard A. Olshen. 1984. *Classification and Regression Trees*. CRC Press.

[10] Jenna Burrell. 2016. How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society* 3, 1 (2016), 12 pages. https://doi.org/10.1177/2053951715622512

[11] Victor R Basili1 Gianluigi Caldiera and H Dieter Rombach. 1994. The goal question metric approach. In *Encyclopedia of software engineering*, John J. Marciniak (Ed.). John Wiley & Sons, Inc., 528–532.

[12] Roberta Calegari, Giovanni Ciatto, and Andrea Omicini. 2020. On the integration of symbolic and sub-symbolic techniques for XAI: A survey. *Intelligenza Artificiale* 14, 1 (2020), 7–32. https://doi.org/10.3233/IA-190036

[13] Davide Calvaresi, Giovanni Ciatto, Amro Najjar, Reyhan Aydoğan, Leon Van der Torre, Andrea Omicini, and Michael Schumacher. 2021. Expectation: Personalized Explainable Artificial Intelligence for Decentralized Agents with Heterogeneous Knowledge. In *Explainable and Transparent AI and Multi-Agent Systems. Third International Workshop, EXTRAAMAS 2021, Virtual Event, May 3–7, 2021, Revised Selected Papers*, Davide Calvaresi, Amro Najjar, Michael Winikoff, and Kary Främling (Eds.). LNCS, Vol. 12688. Springer Nature, Basel, Switzerland, 331–343. https://doi.org/10.1007/978-3-030-82017-6_20

[14] Giovanni Ciatto, Roberta Calegari, Andrea Omicini, and Davide Calvaresi. 2019. Towards XMAS: eXplainability through Multi-Agent Systems. In *AI&IoT 2019 – Artificial Intelligence and Internet of Things 2019 (CEUR Workshop Proceedings, Vol. 2502)*, Claudio Savaglio, Giancarlo Fortino, Giovanni Ciatto, and Andrea Omicini (Eds.). CEUR-WS.org, Rende, CS, Italy, 40–53. http://ceur-ws.org/Vol-2502/paper3.pdf

[15] Giovanni Ciatto, Davide Calvaresi, Michael I. Schumacher, and Andrea Omicini. 2020. An Abstract Framework for Agent-Based Explanations in AI. In *19th International Conference on Autonomous Agents and MultiAgent Systems* (Auckland, New Zeland), Amal El Fallah Seghrouchni, Gita Sukthankar, Bo An, and Neil Yorke-Smith (Eds.). IFAAMAS, Auckland, New Zeland, 1816–1818. http://ifaamas.org/Proceedings/aamas2020/pdfs/p1816.pdf

[16] Giovanni Ciatto, Michael I. Schumacher, Andrea Omicini, and Davide Calvaresi. 2020. Agent-Based Explanations in AI: Towards an Abstract Framework. In *Explainable, Transparent Autonomous Agents and Multi-Agent Systems*, Davide Calvaresi, Amro Najjar, Michael Winikoff, and Kary Främling (Eds.). LNCS, Vol. 12175. Springer, Cham, Auckland,

New Zealand, 3–20. https://doi.org/10.1007/978-3-030-51924-7_1

[17] Philipp Cimiano. 2006. *Ontology Learning and Population from Text.* Springer US. https://doi.org/10.1007/978-0-387-39252-3

[18] Artur S. d'Avila Garcez, Krysia Broda, and Dov M. Gabbay. 2001. Symbolic knowledge extraction from trained neural networks: A sound approach. *Artif. Intell.* 125, 1-2 (2001), 155–207. https://doi.org/10.1016/S0004-3702(00)00077-1

[19] Alex A. Freitas. 2014. Comprehensible classification models: a position paper. *ACM SIGKDD Explorations Newsletter* 15, 1 (June 2014), 1–10. https://doi.org/10.1145/2594473.2594475

[20] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. 2016. *Deep Learning.* MIT Press. http://www.deeplearningbook.org/

[21] Bryce Goodman and Seth Flaxman. 2017. European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation". *AI Magazine* 38, 3 (2017), 50–57. https://doi.org/10.1609/aimag.v38i3.2741

[22] Marco Gori (Ed.). 2018. *Machine Learning: A Constraint Based Approach.* Morgan Kaufmann. 560 pages. https://doi.org/10.1016/C2015-0-00237-4

[23] C. Cordell Green and Bertram Raphael. 1968. The use of theorem-proving techniques in question-answering systems. In *Proceedings of the 23rd ACM national conference (ACM 1968)*, Richard B. Blue Sr. and Arthur M. Rosenberg (Eds.). ACM, New York, NY, USA, 169–181. https://doi.org/10.1145/800186.810578

[24] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A Survey of Methods for Explaining Black Box Models. *Comput. Surveys* 51, 5 (2018), 1–42. https://doi.org/10.1145/3236009

[25] Tameru Hailesilassie. 2016. Rule Extraction Algorithm for Deep Neural Networks: A Review. *CoRR* abs/1610.05267 (2016), 1–6. arXiv:1610.05267 http://arxiv.org/abs/1610.05267

[26] Alfred Horn. 1951. On sentences which are true of direct unions of algebras. *Journal of Symbolic Logic* 16, 1 (1951), 14–21. https://doi.org/10.2307/2268661

[27] Johan Huysmans, Bart Baesens, and Jan Vanthienen. 2006. *Using Rule Extraction to Improve the Comprehensibility of Predictive Models.* K.U. Leuven KBI Working Paper 0612. Katholieke Universiteit Leuven. https://ssrn.com/abstract=961358

[28] Johan Huysmans, Karel Dejaeger, Christophe Mues, Jan Vanthienen, and Bart Baesens. 2011. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems* 51, 1 (2011), 141–154. https://doi.org/10.1016/j.dss.2010.12.003

[29] Sotiris B. Kotsiantis. 2007. Supervised Machine Learning: A Review of Classification Techniques. *Informatica (Slovenia)* 31, 3 (2007), 249–268. http://www.informatica.si/index.php/informatica/article/view/148

[30] Luís C. Lamb, Artur S. d'Avila Garcez, Marco Gori, Marcelo O. R. Prates, Pedro H. C. Avelar, and Moshe Y. Vardi. 2020. Graph Neural Networks Meet Neural-Symbolic Computing: A Survey and Perspective. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, Christian Bessiere (Ed.). ijcai.org, Yokohama, Japan, 4877–4884. https://doi.org/10.24963/ijcai.2020/679

[31] Hector J. Levesque and Ronald J. Brachman. 1987. Expressiveness and tractability in knowledge representation and reasoning. *Comput. Intell.* 3 (1987), 78–93. https://doi.org/10.1111/j.1467-8640.1987.tb00176.x

[32] Zachary C. Lipton. 2018. The Mythos of Model Interpretability. *Queue* 16, 3 (June 2018), 31–57. https://doi.org/10.1145/3236386.3241340

[33] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan L. Yuille, Jonathan Huang, and Kevin Murphy. 2018. Progressive Neural Architecture Search. In *Proceedings of the 15th European Conference on Computer Vision (ECCV 2018) – Part I (Lecture Notes in Computer Science, Vol. 11205)*, Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (Eds.). Springer, Munich, Germany, 19–35. https://doi.org/10.1007/978-3-030-01246-5_2

[34] John W. Lloyd (Ed.). 1990. *Computational Logic.* Springer. https://doi.org/10.1007/978-3-642-76274-1

[35] Jocelyn Maclure. 2020. The new AI spring: a deflationary view. *AI Soc.* 35, 3 (2020), 747–750. https://doi.org/10.1007/s00146-019-00912-z

[36] J. A. Makowsky. 1987. Why horn formulas matter in computer science: Initial structures and generic examples. *J. Comput. System Sci.* 34, 2 (1987), 266–292. https://doi.org/10.1016/0022-0000(87)90027-4

[37] George F. McNulty. 1977. Fragments of first order logic, I: Universal Horn logic. *Journal of Symbolic Logic* 42, 2 (1977), 221–237. https://doi.org/10.2307/2272123

[38] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artif. Intell.* 267 (2019), 1–38. https://doi.org/10.1016/j.artint.2018.07.007

[39] Thomas M. Mitchell. 1997. *Machine Learning* (1 ed.). McGraw-Hill, Inc., New York, NY, USA.

[40] Patrick M. Murphy and Michael J. Pazzani. 1991. ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees. In *Machine Learning Proceedings 1991.* Elsevier, 183–187.

[41] Ali Bou Nassif, Ismail Shahin, Imtinan Basem Attili, Mohammad Azzeh, and Khaled Shaalan. 2019. Speech Recognition Using Deep Neural Networks: A Systematic Review. *IEEE Access* 7 (2019), 19143–19165. https://doi.org/10.1109/

ACCESS.2019.2896880

[42] Daniel W. Otter, Julian R. Medina, and Jugal K. Kalita. 2021. A Survey of the Usages of Deep Learning for Natural Language Processing. *IEEE Trans. Neural Networks Learn. Syst.* 32, 2 (2021), 604–624. https://doi.org/10.1109/TNNLS.2020.2979670

[43] J. Ross Quinlan. 1987. Simplifying decision trees. *International Journal of Man-Machine Studies* 27, 3 (1987), 221–234. https://doi.org/10.1016/S0020-7373(87)80053-6

[44] J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kauffmann, San Mateo, CA, USA. https://dl.acm.org/doi/10.5555/152181

[45] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1, 5 (2019), 206–215. https://doi.org/10.1038/s42256-019-0048-x

[46] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. 2021. Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges. *CoRR* abs/2103.11251 (2021). arXiv:2103.11251 https://arxiv.org/abs/2103.11251

[47] D. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323 (1986), 533–536. https://doi.org/10.1038/323533a0

[48] Gabriele Tolomei, Fabrizio Silvestri, Andrew Haines, and Mounia Lalmas. 2017. Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'17)*. ACM Press, Halifax, NS, Canada, 465–474. http://dl.acm.org/citation.cfm?id=3098039

[49] Bhekisipho Twala. 2010. Multiple classifier application to credit risk assessment. *Expert Systems with Applications* 37, 4 (2010), 3326–3336.

[50] Johan Van Benthem and Kees Doets. 2001. Higher-Order Logic. In *Handbook of Philosophical Logic*, D. M. Gabbay and F. Guenthner (Eds.). Springer Netherlands, Dordrecht, 189–243. https://doi.org/10.1007/978-94-015-9833-0_3

[51] Tim van Gelder. 1990. Why Distributed Representation is Inherently Non-Symbolic. In *Konnektionismus in Artificial Intelligence und Kognitionsforschung (Informatik-Fachberichte, Vol. 252)*, Georg Dorffner (Ed.). Springer, Salzburg, Österreich, Austria, 58–66. https://doi.org/10.1007/978-3-642-76070-9_6

[52] F. Van Veen and S. Leijnen. 2019. The Neural Network Zoo. https://www.asimovinstitute.org/neural-network-zoo. [Online; accessed 17-September-2021].

[53] Paul Voigt and Axel von dem Bussche. 2017. *The EU General Data Protection Regulation (GDPR). A Practical Guide*. Springer. https://doi.org/10.1007/978-3-319-57959-7

[54] Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Michal Walczak, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker. 2021. Informed Machine Learning – A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems. *IEEE Transactions on Knowledge and Data Engineering* 35, 1 (2021), 614–633. https://doi.org/10.1109/TKDE.2021.3079836

[55] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Trans. Knowl. Data Eng.* 29, 12 (2017), 2724–2743. https://doi.org/10.1109/TKDE.2017.2754499

[56] David H. Wolpert and William G. Macready. 1997. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* 1, 1 (1997), 67–82. https://doi.org/10.1109/4235.585893

[57] Yaqi Xie, Ziwei Xu, Kuldeep S. Meel, Mohan S. Kankanhalli, and Harold Soh. 2019. Embedding Symbolic Knowledge into Deep Networks. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). Curran Associates, Inc., Vancouver, BC, Canada, 4235–4245. https://proceedings.neurips.cc/paper/2019/hash/7b66b4fd401a271a1c7224027ce111bc-Abstract.html

[58] Cheng Yang, Jiawei Liu, and Chuan Shi. 2021. Extract the Knowledge of Graph Neural Networks and Go Beyond it: An Effective Knowledge Distillation Framework. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 1227–1237. https://doi.org/10.1145/3442381.3450068

[59] Ying Yang, Geoffrey I. Webb, and Xindong Wu. 2010. Discretization Methods. In *Data Mining and Knowledge Discovery Handbook, 2nd ed*, Oded Maimon and Lior Rokach (Eds.). Springer, 101–116. https://doi.org/10.1007/978-0-387-09823-4_6

[60] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. 2019. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Networks Learn. Syst.* 30, 11 (2019), 3212–3232. https://doi.org/10.1109/TNNLS.2018.2876865

[61] Jan Ruben Zilke, Eneldo Loza Mencía, and Frederik Janssen. 2016. DeepRED – Rule Extraction from Deep Neural Networks. In *Discovery Science (Lecture Notes in Computer Science, Vol. 9956)*, Toon Calders, Michelangelo Ceci, and Donato Malerba (Eds.). Springer, Bari, Italy, 457–473. https://doi.org/10.1007/978-3-319-46307-0_29

Table A. Example of decision table with 3 binary input variables and 1 output feature—namely, the class label. "Rule set #1" and "Rule set #2" are semantically equivalent, but the latter is more compact since it exploits the *don't care* symbol (–).

Variables	**Rule set #1**								**Rule set #2**			
X	0	0	0	0	1	1	1	1	–	0	1	1
Y	0	0	1	1	0	0	1	1	1	0	0	0
Z	0	1	0	1	0	1	0	1	–	–	0	1
Output	B	B	A	A	B	C	A	A	A	B	B	C

SUPPLEMENTARY MATERIAL

Decision Tables, Trees and Rule Lists

Decision tables are concise visual representations of rules, specifying one or more conclusions for each set of different conditions. They have a column for each input and output variable, and a row for each rule. Each cell c_{ij} contains the value of the j-th variable for the i-th rule. An example of decision table is provided in Table A.

Notably, rule lists, decision trees, and decision tables are essentially equivalent in their theoretical expressiveness. Differences mostly lay in their syntax, i.e. how they represent rules. Conversions among these forms are possible, in the general case. For instance, the decision table from Table A can be converted into the rule tree depicted in Figure A, as well as into the following rule list:

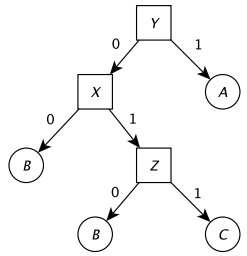


Fig. A. Decision tree equivalent to the decision table reported in Table A.

$$\text{if } Y = 1 \text{ then } A \text{ else if } Y = 0 \wedge X = 1 \wedge Z = 1 \text{ then } C \text{ else } B.$$

Summary about SKE

Table B summarises our analysis regarding the 129 surveyed SKE methods. Notably, the table enumerates SKE methods in chronological order (w.r.t. publication year), grouping them by five-year periods. Furthermore, coherently w.r.t. the sections above, the table reports the translucency, the required task and input type, and the output format and shape of each surveyed method.

Table B. Summary of the knowledge-extraction algorithms. Values from the columns "Translucency", "Task", "Input", "Expressiveness", and "Shape" refer the corresponding figures form Sec. 4.1. Column "Tech." reports the availability/lack of some software technology implementing the algorithm. There, 'L' denotes the presence of a reusable sofware library, whereas 'E' denotes the presence of experimental code, and '?' denotes lack of known technologies.

#	Method	Year	Trans.	Task	Input	Express.	Shape	Tech.
1	Breiman et al. [29]	1984	P	C+R	C+D	P	DT	L[1]
2	Quinlan [163]	1986	P	C	D	P	DT	E[1]
3	Saito and Nakano [175]	1988	P	C	D	P	L	?

[1] https://scikit-learn.org/stable/modules/tree.html

Table B. Summary of the knowledge-extraction algorithms (Continued).

#	Method	Year	Trans.	Task	Input	Express.	Shape	Tech.
4	Clark and Niblett [47]	1989	P	C	C+D	P	L	E[2]
5	Masuoka et al. [141]	1990	D (ANN)	C	C	F	L	?
6	Hayashi [90]	1990	D (ANN)	C	B	F	L	?
7	Towell and Shavlik [206]	1991	D (ANN)	C	D	MN	L	?
8	Berenji [19]	1991	D (ANN)	C	C	F	L	?
9	Brunk and Pazzani [31]	1991	P	C	C+D	P	L	?
10	Murphy and Pazzani [146]	1991	P	C	D	MN	DT	?
11	Horikawa et al. [97]	1992	D (ANN)	C	C	F	L	?
12	Tresp et al. [210]	1992a	D (ANN)	R	C	P	L	?
13	Towell and Shavlik [207]	1993	D (ANN)	C	D	P	L	?
14	Thrun [203]	1993	D (ANN)	C	C	P+MN	L	?
15	Cohen [48]	1993	P	C	C+D	P	L	?
16	Quinlan [164]	1993	P	C	C+D	P	DT	L[3]
17	Fu [77]	1994	D (ANN)	C	D	P	L	?
18	Halgamuge and Glesner [89]	1994	D (ANN)	C	C	F	L	?
19	Mitra [144]	1994	D (ANN)	C	C+D	F	L	?
20	Craven and Shavlik [50]	1994	P	C	B	P+MN	L	?
21	Fürnkranz and Widmer [80]	1994	P	C	D	P	L	E[4]
22	Sestito and Dillon [183]	1994	P	C	C	P	L	?
23	Andrews and Geva [3]	1995	D (ANN)	C	C+D	P	L	?
24	Matthews and Jagielska [142]	1995	D (ANN)	C	B	F	L	?
25	Cohen [49]	1995	P	C	C+D	P	L	L[5]
26	Pop et al. [161]	1994	P	C	B	P	L	?
27	Setiono and Liu [190]	1996	D (ANN)	C	B	P	L	?
28	Tickle et al. [204]	1996	P	C	B	P	L	?
29	Yuan and Zhuang [235]	1996	P	C	D	F	L	?

[2]https://github.com/alessiamondolo/cn2-rule-based-classifier
[3]https://en.wikipedia.org/wiki/C4.5_algorithm#Implementations
[4]https://github.com/buoto/irep-rule-induction
[5]https://github.com/imoscovitz/wittgenstein

Table B. Summary of the knowledge-extraction algorithms (Continued).

#	Method	Year	Trans.	Task	Input	Express.	Shape	Tech.
30	Craven and Shavlik [51]	1996	P	C	B	P+MN	DT	E[6]
31	Hong and Lee [96]	1996	P	C	C	F	L	?
32	Setiono and Liu [191]	1997	D (ANN3)	C	C+D	O	L	?
33	Setiono [185]	1997	D (ANN)	C	D	P	L	?
34	Nauck and Kruse [147]	1997	D (ANN)	C	D	F	L	?
35	Saito and Nakano [176]	1997	D (ANN)	R	C	P	L	?
36	Benítez et al. [18]	1997	D (ANN)	C+R	C	F	L	?
37	Ishibuchi et al. [103]	1997	P	C	C	F	L	?
38	Taha and Ghosh [200]	1999	D (ANN)	C	C	P	L	?
39	Taha and Ghosh [200]	1999	D (ANN)	C	C	P	L	?
40	Krishnan et al. [114]	1999b	D (ANN)	C	B	P	L	?
41	Nauck and Kruse [148]	1999	D (ANN)	R	D	F	L	E[7]
42	Taha and Ghosh [200]	1999	P	C	B	P	L	?
43	Krishnan et al. [113]	1999a	P	C	C	P	DT	?
44	Schmitz et al. [181]	1999	P	C+R	C+D	P	DT	L[8]
45	Hong and Chen [94]	1999	P	C	C	F	L	?
46	Setiono [186]	2000	D (ANN)	C	B	MN	L	?
47	Tsukimoto [213]	2000	D (ANN)	C	C+D	P	L	?
48	Kim and Lee [109]	2000	D (ANN4)	C	C+D	P	DT	?
49	Setiono and Leow [188]	2000	D (ANN)	R	C+D	P+MN+O	DT	?
50	Zhou et al. [240]	2000	P	C	C+D	P	L	?
51	Hong and Chen [95]	2000	P	C	C	F	L	?
52	Sato and Tsukimoto [179]	2001	D (ANN3)	R	C+D	P	DT	E[9]
53	Parpinelli et al. [156]	2001	P	C	C+D	P	L	L[10]
54	Castillo et al. [32]	2001	P	C+R	C+D	F	L	?
55	Saito and Nakano [177]	2002	D (ANN)	R	C+D	P	L	?
56	Setiono et al. [189]	2002	D (ANN3)	R	C+D	P	L	?
57	Liu et al. [123]	2002	P	C	C+D	P	L	?

[6]https://github.com/abarthakur/trepan_python
[7]http://fuzzy.cs.ovgu.de/nefprox/
[8]https://github.com/fantamat/ruleex
[9]https://github.com/zju-vipa/awesome-neural-trees
[10]https://github.com/febo/myra

Table B. Summary of the knowledge-extraction algorithms (Continued).

#	Method	Year	Trans.	Task	Input	Express.	Shape	Tech.
58	Boz [28]	2002	P	C	C+D	P	DT	?
59	Markowska-Kaczmar and Trelak [136]	2003	P	C	C+D	F	L	?
60	Zhou et al. [241]	2003	P	C	C+D	P	L	?
61	Setiono and Thong [192]	2004	D (ANN3)	R	C+D	P	L	?
62	Fu et al. [78]	2004	D (SVM)	C	C+D	P	L	?
63	Markowska-Kaczmar and Chumieja [135]	2004	P	C	C+D	P	L	?
64	Rabuñal et al. [165]	2004	P	C	C+D	P	L	?
65	Chen [41]	2004	P	C	C	P	L	?
66	Liu et al. [124]	2004	P	C	C+D	P	L	E[11]
67	Browne et al. [30]	2004	P	C	C+D	P+MN	DT	?
68	Zhang et al. [236]	2005	D (SVM)	C	C	P	L	?
69	Barakat and Diederich [13]	2008	D (SVM)	C+R	C	P	DT	?
70	Fung et al. [79]	2005	D (LC)	C	C	P	L	?
71	Chaves et al. [39]	2005	D (SVM)	C	C	F	L	?
72	Torres and Rocco [205]	2005	P	C	C+D	P+MN	DT	?
73	Etchells and G. [69]	2006	P	C	C+D	P	L	?
74	He et al. [92]	2006	P	C	C+D	P	DT	?
75	Huysmans et al. [101]	2006	P	R	C	P	L	L[12]
76	Bader et al. [8]	2007	D (ANN)	C	B	P	L	?
77	Schetinin et al. [180]	2007	D (DTE)	R	C	P	DT	?
78	Chen et al. [43]	2007	D (SVM)	C	C	P	L	?
79	Barakat and Bradley [12]	2007	D (SVM)	C	C+D	P	L	?
80	Saad and Wunsch II [171]	2007	P	C	C+D	O	L	E[8]
81	Martens et al. [140]	2007	P	C	C+D	P	L	?
82	Núñez et al. [152]	2008	D (SVM)	C	C	P+O	L	?
83	Setiono et al. [187]	2008	P	C	C+D	P+O	L	?
84	Odajima et al. [154]	2008	P	C	D	P	L	?
85	Konig et al. [112]	2008	P	C+R	C+D	F	DT	?
86	Bader [5]	2009	D (ANN)	C	B	P	L	?
87	Martens et al. [139]	2009	D (SVM)	C	C	any	any	?
88	Lehmann et al. [115]	2010	P	C	B	P	L	?

[11] https://rdrr.io/github/adriansidor/antminer/src/R/antminer3.R
[12] https://github.com/psykei/psyke-python

Table B. Summary of the knowledge-extraction algorithms (Continued).

#	Method	Year	Trans.	Task	Input	Express.	Shape	Tech.
89	Augasta and Kathirvalavakumar [4]	2012	P	C	C+D	P	L	?
90	Sethi et al. [184]	2012	P	C	C+D	P	TA	?
91	Zilke et al. [243]	2016	D (ANN)	R	C+D	P	DT	E[8]
92	Chan and Chan [34]	2017	D (ANN)	R	C	P	L	?
93	Biswas et al. [21]	2017	D (ANN)	C	C	P	L	?
94	Yedjour and Benyettou [232]	2018	P	C	B	P	L	?
95	Chan and Chan [35]	2020	D (ANN)	R	C	P	L	?
96	Wang et al. [217]	2020	D (DTE)	C	C	P	L	?
97	Chen et al. [42]	2020	D (ANN)	C	I	P	L	E[13]
98	Mahdavifar and Ghorbani [130]	2020	D (ANN)	C	B	P	L	?
99	Vasilev et al. [214]	2020	D (ANN)	C	C	P	L	?
100	Odense and d'Avila Garcez [155]	2020	D (ANN)	C	I	MN	L	?
101	Jia et al. [106]	2020	D (ANN)	C	I	P	DT	?
102	Li et al. [116]	2020	D (ANN)	C	C	F	L	?
103	Hayashi and Takano [91]	2020	D (ANN)	C	C+D	P	L	?
104	Chakraborty et al. [33]	2020	D (ANN)	C	C+D	P	L	?
105	Sabbatini et al. [174]	2021	P	R	C	P	L	E[12]
106	Yu and Liu [234]	2021	D (ANN)	C	C	P	L	?
107	Yan et al. [228]	2021	D (ANN)	C	C	P	DT	?
108	Dattachaudhuri et al. [57]	2021	D (ANN)	C	C	P	L	?
109	Dong et al. [66]	2021	P	C	C+D	P	L	?
110	Shams et al. [193]	2021	D (ANN)	C	C	P	L	L[14]
111	Yedjour [231]	2021	P	C	C	P	L	?
112	Marshakov [138]	2021	D (ANN)	C	C	F	L	?
113	Yang et al. [230]	2021	D (GNN)	C	G	KG	T	E[15]
114	Bastos et al. [17]	2021	D (GNN)	C	T	KG	T	E[16]
115	Horta et al. [98]	2021	D (ANN)	C	I	KG	T	?
116	Bologna [22]	2021	D (DTE)	C	C	P	L	?
117	Espinosa Zarlenga et al. [68]	2021	D (ANN)	C	C	P	L	E[14]
118	Sabbatini and Calegari [172]	2022	P	R	C	P	L	E[12]

[13] https://github.com/SeekingDream/FSE20_DENAS
[14] https://github.com/mateoespinosa/remix
[15] https://github.com/BUPT-GAMMA/CPF
[16] https://github.com/ansonb/RECON

Table B. Summary of the knowledge-extraction algorithms (Continued).

#	Method	Year	Trans.	Task	Input	Express.	Shape	Tech.
119	Johansson et al. [108]	2022	P	R	C	P	L	?
120	Barbiero et al. [15]	2022	D (ANN)	C	I	P	L	L[17]
121	Ferreira et al. [74]	2022	D (ANN)	C	I	P	L	?
122	Diao et al. [62]	2022	D (ANN4)	C	C	P	L	?
123	Barbado et al. [14]	2022	D (SVM)	C	C+D	P	L	L[18]
124	de Campos Souza and Lughofer [60]	2022	D (ANN3)	C	C	P	L	?
125	Salimi-Badr and Ebadzadeh [178]	2022	D (ANN)	R	C	P	L	?
126	Irfan et al. [102]	2022	D (CNN)	C	I	P	L	?
127	Sabbatini and Calegari [173]	2023	P	C+R	C	P	DT	E[12]
128	Obregon and Jung [153]	2023	D (DTE)	C	C+D	P	L	E[19]
129	Ciravegna et al. [46]	2023	D (ANN)	C	B	P	L	L[20]

Summary about SKI

Table C summarises our analysis regarding the 117 surveyed SKI methods. Notably, the table enumerates SKI methods in chronological order (w.r.t. publication year), grouping them by five-year periods. Furthermore, coherently w.r.t. the sections above, the table reports the strategy followed by each SKI method, as well as type of knowledge it can inject, the type of neural network it supports, and overall purpose it supports injection for.

Table C. Summary of knowledge-injection algorithms. Values from the columns "Strategy", "Input", "Predictor", and "Purpose" refer the corresponding figures form Sec. 4.2. Column "Tech." reports the availability/lack of some software technology implementing the algorithm. There, 'L' denotes the presence of a reusable sofware library, whereas 'E' denotes the presence of experimental code, and '?' denotes lack of known technologies.

#	Method	Year	Strategy	Input	Predictor	Purpose	Tech.
1	Ballard [11]	1986	S	FOL	BM	M	?
2	Towell et al. [208]	1990	S	P	FF	E	L[21]
3	Pinkus [160]	1991	S	FOL	BM	M	?
4	Tresp et al. [211]	1992b	L+S	P	FF	E+M	?
5	Giles and Omlin [83]	1993	S	E	RNN	E+M	?
6	Tan [201]	1997	S	P	FF	E	?
7	d'Avila Garcez and Zaverucha [59]	1999	S	P	FF	M	L[22]
8	Basilio et al. [16]	2001	L+S	FOL	FF	M	?

[17] https://github.com/pietrobarbiero/pytorch_explain
[18] https://github.com/AlbertoBarbado/rule_extraction_xai
[19] https://github.com/jobregon1212/rulecosi
[20] https://github.com/pietrobarbiero/logic_explained_networks
[21] https://github.com/psykei/psyki-python
[22] https://sourceforge.net/projects/cil2p/

Table C. Summary of knowledge-injection algorithms (continued).

#	Method	Year	Strategy	Input	Predictor	Purpose	Tech.
9	d'Avila Garcez and Gabbay [58]	2004	S	FOL	FF	M	?
10	Bader et al. [6]	2005	S	FOL	FF	M	?
11	Chang et al. [37]	2007	E	E	MN	E	?
12	Bader et al. [7]	2008	L	E	FF	E	?
13	Mintz et al. [143]	2009	E	KG	FF	M	?
14	Nickel et al. [151]	2011	E	KG	FF	M	L[23]
15	Bordes et al. [26]	2011	E	KG	FF	M	E[24]
16	Kimmig et al. [110]	2012	S	FOL	MN	M	L[25]
17	Bordes et al. [23]	2012	E	KG	FF	M	?
18	Pinkas et al. [159]	2013	S	FOL	BM	M	?
19	Bordes et al. [25]	2013	E+L	KG	FF	M	E[26]
20	Socher et al. [195]	2013	E+S	KG	FF	M	E[27]
21	França et al. [76]	2014	S	P	RNN	M	E[28]
22	Wang et al. [218]	2014	E+L	KG	FF	M	E[29]
23	García-Durán et al. [81]	2014	E+L	KG	FF	M	?
24	Bian et al. [20]	2014	E+L	E	FF	E	?
25	Chang et al. [36]	2014	E	KG	FF	M	?
26	Bordes et al. [24]	2014	E	KG	FF	M	E[30]
27	Dong et al. [67]	2014	E	KG	FF	M	?
28	Fan et al. [71]	2014	E+L	KG	FF	M	?
29	Wang et al. [216]	2015	E	KG	FF	M	?
30	Wei et al. [220]	2015	E	KG	MN	M	E[31]
31	Rocktäschel et al. [169]	2015	E+L	KG	FF	M	E[32]
32	Lin et al. [122]	2015	E+L	KG	FF	M	E[33]
33	Yang et al. [229]	2015	E+L	KG	FF	M	?
34	Che et al. [40]	2015	L	KG	FF	E	?
35	Ji et al. [104]	2015	E+L	KG	FF	M	?
36	Feng et al. [73]	2016	E+L	KG	FF	M	?
37	Xiao et al. [223]	2015	E+L	KG	FF	M	?
38	He et al. [93]	2015	E+L	KG	FF	M	?
39	Tran and Garcez [209]	2016	S	P	DBN	E	?
40	Hu et al. [99]	2016a	S	P	CNN	E	?

[23]https://github.com/mnick/rescal.py
[24]https://github.com/glorotxa/SME
[25]https://github.com/linqs/psl
[26]https://github.com/Lapis-Hong/TransE-Knowledge-Graph-Embedding
[27]https://github.com/dddoss/tensorflow-socher-ntn
[28]https://github.com/vakker/CILP
[29]https://github.com/thunlp/KB2E
[30]https://github.com/usherwang02/SemanticMatchingEnergy-Theano
[31]https://github.com/ZhuoyuWei/fpMLN
[32]https://github.com/uclnlp/low-rank-logic
[33]https://github.com/thunlp/KB2E

Table C. Summary of knowledge-injection algorithms (continued).

#	Method	Year	Strategy	Input	Predictor	Purpose	Tech.
41	Guo et al. [88]	2016	E+L	KG	FF	M	?
42	Nickel et al. [150]	2016	E+L	KG	FF	M	E[34]
43	Trouillon et al. [212]	2016	E+L	KG	FF	M	E[35]
44	Demeester et al. [61]	2016	L	KG	FF	M	?
45	Hu et al. [100]	2016b	S	P	FF	E	?
46	Mrksic et al. [145]	2016	L	KG	FF	E	E[36]
47	Liu et al. [126]	2016	E	KG	FF	M	?
48	Ji et al. [105]	2016	E+L	KG	FF	M	?
49	Xiao et al. [225]	2016b	E+L	KG	FF	M	?
50	Xiao et al. [224]	2016a	E+L	KG	FF	M	?
51	Kipf and Welling [111]	2017	E+L	KG	GNN	M	L[37]
52	Rocktäschel and Riedel [168]	2017	L+S	D	FF	M	E[38]
53	Liu et al. [125]	2017	E+L	KG	FF	M	E[39]
54	Stewart and Ermon [198]	2017	L	E	CNN	E	?
55	Allamanis et al. [2]	2017	L	P	RNN	E	E[40]
56	Diligenti et al. [63]	2017a	L	FOL	KM	M	E[41]
57	Diligenti et al. [64]	2017b	L	P	CNN	M	?
58	Marino et al. [134]	2017	E	KG	GNN	E	?
59	Chang et al. [38]	2017	E	E	FF	E	E[42]
60	Choi et al. [45]	2017	E	KG	FF	E	E[43]
61	Fang et al. [72]	2017	L	KG	CNN	E	?
62	Xu et al. [227]	2018	L	P	CNN	E	E[44]
63	Evans and Grefenstette [70]	2018	L+S	D	FF	M	E[45]
64	Sourek et al. [196]	2018	S	D	FF	M	E[46]
65	Velickovic et al. [215]	2018	E+L	KG	GNN	M	E[47]
66	Ma and Zhang [127]	2018	L	KG	AE	E	?
67	Zhou et al. [239]	2018	E	KG	GNN	E	E[48]
68	Liang et al. [120]	2018	S	KG	FF	E	E[49]

[34]https://github.com/mnick/holographic-embeddings
[35]https://github.com/thunlp/openke
[36]https://github.com/nmrksic/counter-fitting
[37]https://github.com/tkipf/pygcn
[38]https://github.com/uclnlp/ntp
[39]https://github.com/quark0/ANALOGY
[40]https://github.com/mast-group/eqnet
[41]https://sites.google.com/site/semanticbasedregularization/home/software
[42]https://github.com/mbchang/dynamics
[43]https://github.com/mp2893/gram
[44]https://github.com/UCLA-StarAI/Semantic-Loss/
[45]https://github.com/crunchiness/lernd
[46]https://github.com/GustikS/GNNwLRNNs
[47]https://github.com/PetarV-/GAT
[48]https://github.com/tuxchow/ccm
[49]https://github.com/julianschoep/SGRLayer

Table C. Summary of knowledge-injection algorithms (continued).

#	Method	Year	Strategy	Input	Predictor	Purpose	Tech.
69	Glavas and Vulic [85]	2018	E+L	KG	FF	E	E[50]
70	Marra et al. [137]	2019	L	P	FF	E	E[51]
71	Goodwin and Demner-Fushman [86]	2019	L	KG	FF	E	?
72	Sun et al. [199]	2019	E+L	KG	FF	M	?
73	Zhang et al. [238]	2019	L	KG	TR	E	?
74	Peters et al. [158]	2019	E+L	KG	TR	E	L[52]
75	Daniele and Serafini [54]	2019	S	FOL	DFF	E	L[53]
76	Fischer et al. [75]	2019	L	D	DFF	E	E[54]
77	Dong et al. [65]	2019	S+L	H	FF	M	E[55]
78	Badreddine et al. [9]	2020	S	FOL	FF	E+M	L[56]
79	Zhang et al. [237]	2020	E+L	KG	FF	M	E[57]
80	Jiang et al. [107]	2020	S+L	FOL	RNN	E	?
81	Ren and Leskovec [166]	2020	S+L	KG	DFF	M	E[58]
82	Guo et al. [87]	2020	L+E	KG	FF	M	E[59]
83	Riegel et al. [167]	2020	S+L	FOL	FF	M	L[60]
84	Yu and Liu [234]	2021	S	P	DAE	E	?
85	Manhaeve et al. [131]	2021	S	H	FF	E+M	L[61]
86	Dash et al. [56]	2021	E	P	GNN	E	E[62]
87	Giunchiglia and Lukasiewicz [84]	2021	S+L	P	CNN	E	E[63]
88	Bosselut et al. [27]	2021	S	KG	TR	M	?
89	Peng et al. [157]	2021	E	E	TR	E	?
90	West et al. [222]	2022	L E	KG	TR	E	?
91	Marino et al. [133]	2021	S+L	KG	TR	E	L[64]
92	Xie et al. [226]	2021	S+L	M	RNN	E	?
93	Cheng et al. [44]	2021	L+E	H	FF	M	?
94	Li et al. [118]	2023a	S+L	D	GNN	M	?
95	d'Amato et al. [53]	2021	L+E	KG	FF	M	E[65]

[50] https://github.com/codogogo/explirefit
[51] https://github.com/GiuseppeMarra/lyrics
[52] https://github.com/allenai/kb
[53] https://github.com/DanieleAlessandro/KENN
[54] https://github.com/eth-sri/dl2
[55] https://github.com/google/neural-logic-machines
[56] https://github.com/logictensornetworks/logictensornetworks
[57] https://github.com/MIRALab-USTC/KGE-HAKE
[58] https://github.com/snap-stanford/KGReasoning
[59] https://github.com/StudyGroup-lab/SLRE
[60] https://github.com/IBM/LNN
[61] https://github.com/ML-KULeuven/deepproblog
[62] https://github.com/tirtharajdash/VEGNN
[63] https://github.com/EGiunchiglia/C-HMCNN/
[64] https://github.com/facebookresearch/mmf
[65] https://github.com/Keehl-Mihael/TransROWL-HRS

Table C. Summary of knowledge-injection algorithms (continued).

#	Method	Year	Strategy	Input	Predictor	Purpose	Tech.
96	Dash et al. [55]	2022	S	P	GNN	E	E[66]
97	Rodríguez et al. [170]	2022	L	KG	CNN	E	E[67]
98	Yu et al. [233]	2022	S+L	FOL	CNN	E	?
99	Wei et al. [219]	2022	S+L	P	GNN	M	E[68]
100	Smirnova et al. [194]	2022	L	P	FF	E	E[69]
101	Magnini et al. [128]	2022a	S	D	FF	E	E[21]
102	Spillo et al. [197]	2022	E	FOL	DFF	E	E[70]
103	Tang et al. [202]	2022	L+E	FOL	RNN	M	E[71]
104	Zhu et al. [242]	2022	L	FOL	GNN	M	E[72]
105	Li et al. [117]	2022	L+E	KG	GNN	M	?
106	Sen et al. [182]	2022	S+L	D	FF	M	?
107	Magnini et al. [129]	2022b	S+L	D	DFF	E	E[21]
108	Werner et al. [221]	2023	S	FOL	GNN	E	E[73]
109	Giannini et al. [82]	2023	L	FOL	FF	E	?
110	Cunnington et al. [52]	2023	S+L	D	FF	E	E[74]
111	Pourvali et al. [162]	2023	S+L	FOL	TR	E	?
112	Ahmed et al. [1]	2023	L	P	FF	E	E[75]
113	Marconato et al. [132]	2023	L	H	DFF	M+E	E[76]
114	Li et al. [119]	2023b	S+L	KG	TR	E	E[77]
115	Lin et al. [121]	2023	S+L	H	TR	E	?
116	Bai et al. [10]	2023	L	M	GNN	M	?
117	Nguyen et al. [149]	2023	L+E	FOL	FF	M	E[78]

REFERENCES

[1] Kareem Ahmed, Kai-Wei Chang, and Guy Van den Broeck. 2023. Semantic Strengthening of Neuro-Symbolic Learning. *CoRR* abs/2302.14207 (2023). https://doi.org/10.48550/arXiv.2302.14207 arXiv:2302.14207

[2] Miltiadis Allamanis, Pankajan Chanthirasegaran, Pushmeet Kohli, and Charles Sutton. 2017. Learning Continuous Semantic Representations of Symbolic Expressions. In *Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, NSW, Australia, August 6-11 2017 (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 80–88. http://proceedings.mlr.press/v70/allamanis17a.html

[3] Robert Andrews and Shlomo Geva. 1995. RULEX & CEBP Networks As the Basis for a Rule Refinement System. In *Hybrid Problems, Hybrid Solutions*, J. Hallam (Ed.). IOS Press, 1–12.

[66] https://github.com/tirtharajdash/BotGNN
[67] https://github.com/JulesSanchez/X-NeSyL
[68] http://github.com/jinnanli/CogKG
[69] https://github.com/eXascaleInfolab/Nessy_RE
[70] https://github.com/giuspillo/RepoNeSyRecSys2022
[71] https://github.com/XiaojuanTang/RulE
[72] https://github.com/DeepGraphLearning/GNN-QE
[73] https://gitlab.inria.fr/tyrex-public/kegnn
[74] https://github.com/DanCunnington/FFNSL
[75] https://github.com/UCLA-StarAI/Semantic-Strengthening
[76] https://github.com/ema-marconato/NeSy-CL
[77] https://github.com/senticnet/SKIER
[78] https://github.com/nlp-tlp/cyle

[4] M. Gethsiyal Augasta and T. Kathirvalavakumar. 2012. Reverse Engineering the Neural Networks for Rule Extraction in Classification Problems. *Neural Process. Lett.* 35, 2 (2012), 131–150. https://doi.org/10.1007/s11063-011-9207-8

[5] Sebastian Bader. 2009. Extracting Propositional Rules from Feedforward Neural Networks by Means of Binary Decision Diagrams. In *NeSy'09 – Neural-Symbolic Learning and Reasoning (CEUR Workshop Proceedings, Vol. 481)*, Artur S. d'Avila Garcez and Pascal Hitzler (Eds.). CEUR-WS.org, Pasadena, CA, USA. http://ceur-ws.org/Vol-481/paper-5.pdf

[6] Sebastian Bader, Artur S. d'Avila Garcez, and Pascal Hitzler. 2005. Computing First-Order Logic Programs by Fibring Artificial Neural Networks. In *Proceedings of the 18th International Florida Artificial Intelligence Research Society Conference (FlAIRS), Clearwater Beach, Florida, USA, May 15–17, 2005*, Ingrid Russell and Zdravko Markov (Eds.). AAAI Press, 314–319. http://www.aaai.org/Library/FLAIRS/2005/flairs05-052.php

[7] Sebastian Bader, Steffen Hölldobler, and Nuno C. Marques. 2008. Guiding Backprop by Inserting Rules. In *NeSy'08 – Neural-Symbolic Learning and Reasoning (CEUR Workshop Proceedings, Vol. 366)*, Artur S. d'Avila Garcez and Pascal Hitzler (Eds.). CEUR-WS.org, Patras, Greece. http://ceur-ws.org/Vol-366/paper-5.pdf

[8] Sebastian Bader, Steffen Hölldobler, and Valentin Mayer-Eichberger. 2007. Extracting Propositional Rules from Feed-forward Neural Networks – A New Decompositional Approach. In *NeSy'07 – Neural-Symbolic Learning and Reasoning, 3rd International Workshop (CEUR Workshop Proceedings, Vol. 230)*, Artur S. d'Avila Garcez, Pascal Hitzler, and Guglielmo Tamburrini (Eds.). CEUR-WS.org. http://ceur-ws.org/Vol-230/04-bader.pdf

[9] Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. 2020. Logic Tensor Networks. *CoRR* abs/2012.13635 (2020). arXiv:2012.13635 https://arxiv.org/abs/2012.13635

[10] Luyi Bai, Wenting Yu, Die Chai, Wenjun Zhao, and Mingzhuo Chen. 2023. Temporal knowledge graphs reasoning with iterative guidance by temporal logical rules. *Inf. Sci.* 621 (2023), 22–35. https://doi.org/10.1016/j.ins.2022.11.096

[11] Dana H. Ballard. 1986. Parallel Logical Inference and Energy Minimization. In *Proceedings of the 5th National Conference on Artificial Intelligence. Philadelphia, PA, USA, August 11-15, 1986. Volume 1: Science*, Tom Kehler (Ed.). Morgan Kaufmann, 203–209. http://www.aaai.org/Library/AAAI/1986/aaai86-033.php

[12] Nahla H. Barakat and Andrew P. Bradley. 2007. Rule Extraction from Support Vector Machines: A Sequential Covering Approach. *IEEE Trans. Knowl. Data Eng.* 19, 6 (2007), 729–741. https://doi.org/10.1109/TKDE.2007.190610

[13] Nahla H. Barakat and Joachim Diederich. 2008. Eclectic Rule-Extraction from Support Vector Machines. *International Journal of Computer and Information Engineering* 2, 5 (2008), 1672–1675. https://doi.org/10.5281/zenodo.1055511

[14] Alberto Barbado, Óscar Corcho, and Richard Benjamins. 2022. Rule extraction in unsupervised anomaly detection for model explainability: Application to OneClass SVM. *Expert Syst. Appl.* 189 (2022), 116100. https://doi.org/10.1016/j.eswa.2021.116100

[15] Pietro Barbiero, Gabriele Ciravegna, Francesco Giannini, Pietro Lió, Marco Gori, and Stefano Melacci. 2022. Entropy-Based Logic Explanations of Neural Networks. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 6046–6054. https://ojs.aaai.org/index.php/AAAI/article/view/20551

[16] Rodrigo Basilio, Gerson Zaverucha, and Valmir Carneiro Barbosa. 2001. Learning Logic Programs with Neural Networks. In *Inductive Logic Programming, 11th International Conference, ILP 2001, Strasbourg, France, September 9-11, 2001, Proceedings (Lecture Notes in Computer Science, Vol. 2157)*, Céline Rouveirol and Michèle Sebag (Eds.). Springer, 15–26. https://doi.org/10.1007/3-540-44797-0_2

[17] Anson Bastos, Abhishek Nadgeri, Kuldeep Singh, Isaiah Onando Mulang', Saeedeh Shekarpour, Johannes Hoffart, and Manohar Kaul. 2021. RECON: Relation Extraction using Knowledge Graph Context in a Graph Neural Network. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 1673–1685. https://doi.org/10.1145/3442381.3449917

[18] José Manuel Benítez, Juan Luis Castro, and Ignacio Requena. 1997. Are artificial neural networks black boxes? *IEEE Trans. Neural Networks* 8, 5 (1997), 1156–1164. https://doi.org/10.1109/72.623216

[19] Hamid R. Berenji. 1991. Refinement of Approximate Reasoning-based Controllers by Reinforcement Learning. In *Proceedings of the Eighth International Workshop (ML91), Northwestern University, Evanston, Illinois, USA*, Lawrence Birnbaum and Gregg Collins (Eds.). Morgan Kaufmann, 475–479. https://doi.org/10.1016/b978-1-55860-200-7.50097-0

[20] Jiang Bian, Bin Gao, and Tie-Yan Liu. 2014. Knowledge-Powered Deep Learning for Word Embedding. In *Proceedings of the 25th Machine Learning and Knowledge Discovery in Databases - European Conference (ECML), Nancy, France, September 15-19, 2014 (Lecture Notes in Computer Science, Vol. 8724)*, Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo (Eds.). Springer, 132–148. https://doi.org/10.1007/978-3-662-44848-9_9

[21] Saroj K. Biswas, Manomita Chakraborty, Biswajit Purkayastha, Pinki Roy, and Dalton Meitei Thounaojam. 2017. Rule Extraction from Training Data Using Neural Network. *Int. J. Artif. Intell. Tools* 26, 3 (2017), 1750006:1–1750006:26. https://doi.org/10.1142/S0218213017500063

[22] Guido Bologna. 2021. A Rule Extraction Technique Applied to Ensembles of Neural Networks, Random Forests, and Gradient-Boosted Trees. *Algorithms* 14, 12 (2021), 339. https://doi.org/10.3390/a14120339

[23] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2012. Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2012, La Palma, Canary Islands, Spain, April 21-23, 2012 (JMLR Proceedings, Vol. 22)*, Neil D. Lawrence and Mark A. Girolami (Eds.). JMLR.org, 127–135. http://proceedings.mlr.press/v22/bordes12.html

[24] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data - Application to word-sense disambiguation. *Mach. Learn.* 94, 2 (2014), 233–259. https://doi.org/10.1007/s10994-013-5363-6

[25] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Proceedings of 27th Annual Conference on Neural Information Processing Systems (NeurIPS), Lake Tahoe, Nevada, United States, December 5-8, 2013*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 2787–2795. https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html

[26] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. 2011. Learning Structured Embeddings of Knowledge Bases. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*, Wolfram Burgard and Dan Roth (Eds.). AAAI Press. http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3659

[27] Antoine Bosselut, Ronan Le Bras, and Yejin Choi. 2021. Dynamic Neuro-Symbolic Knowledge Graph Construction for Zero-shot Commonsense Question Answering. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 4923–4931. https://ojs.aaai.org/index.php/AAAI/article/view/16625

[28] Olcay Boz. 2002. Converting A Trained Neural Network To a Decision Tree DecText - Decision Tree Extractor. In *Proceedings of the 2002 International Conference on Machine Learning and Applications - ICMLA 2002, June 24-27, 2002, Las Vegas, Nevada, USA*, M. Arif Wani, Hamid R. Arabnia, Krzysztof J. Cios, Khalid Hafeez, and Graham Kendall (Eds.). CSREA Press, 110–116.

[29] Leo Breiman, Jerome Friedman, Charles J. Stone, and Richard A. Olshen. 1984. *Classification and Regression Trees*. CRC Press.

[30] Antony Browne, Brian D. Hudson, David C. Whitley, Martyn G. Ford, and Philip Picton. 2004. Biological data mining with neural networks: implementation and application of a flexible decision tree extraction algorithm to genomic problem domains. *Neurocomputing* 57 (March 2004), 275–293. https://doi.org/10.1016/j.neucom.2003.10.007

[31] Clifford Brunk and Michael J. Pazzani. 1991. An Investigation of Noise-Tolerant Relational Concept Learning Algorithms. In *Proceedings of the Eighth International Workshop (ML91), Northwestern University, Evanston, Illinois, USA*, Lawrence Birnbaum and Gregg Collins (Eds.). Morgan Kaufmann, 389–393. https://doi.org/10.1016/b978-1-55860-200-7.50080-5

[32] Luis A. Castillo, Antonio González Muñoz, and Raúl Pérez. 2001. Including a simplicity criterion in the selection of the best rule in a genetic fuzzy learning algorithm. *Fuzzy Sets Syst.* 120, 2 (2001), 309–321. https://doi.org/10.1016/S0165-0114(99)00095-0

[33] Manomita Chakraborty, Saroj Kumar Biswas, and Biswajit Purkayastha. 2020. Rule extraction from neural network trained using deep belief network and back propagation. *Knowl. Inf. Syst.* 62, 9 (2020), 3753–3781. https://doi.org/10.1007/s10115-020-01473-0

[34] Veronica Chan and Christine W. Chan. 2017. Towards Developing the Piece-Wise Linear Neural Network Algorithm for Rule Extraction. *Int. J. Cogn. Informatics Nat. Intell.* 11, 2 (2017), 57–73. https://doi.org/10.4018/IJCINI.2017040104

[35] Veronica K.H. Chan and Christine W. Chan. 2020. Towards explicit representation of an artificial neural network model: Comparison of two artificial neural network rule extraction approaches. *Petroleum* 6, 4 (2020), 329–339. https://doi.org/10.1016/j.petlm.2019.11.005 SI: Artificial Intelligence (AI), Knowledge-based Systems (KBS), and Machine Learning (ML).

[36] Kai-Wei Chang, Wen-tau Yih, Bishan Yang, and Christopher Meek. 2014. Typed Tensor Decomposition of Knowledge Bases for Relation Extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, October 25-29, 2014*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). ACL, 1568–1579. https://doi.org/10.3115/v1/d14-1165

[37] Ming-Wei Chang, Lev-Arie Ratinov, and Dan Roth. 2007. Guiding Semi-Supervision with Constraint-Driven Learning. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL), June 23-30, 2007, Prague, Czech Republic*, John A. Carroll, Antal van den Bosch, and Annie Zaenen (Eds.). The Association for Computational Linguistics (ACL). https://aclanthology.org/P07-1036/

[38] Michael Chang, Tomer Ullman, Antonio Torralba, and Joshua B. Tenenbaum. 2017. A Compositional Object-Based Approach to Learning Physical Dynamics. In *Proceedings of the 5th International Conference on Learning Representations (ICLR), Toulon, France, April 24-26, 2017*. OpenReview.net. https://openreview.net/forum?id=Bkab5dqxe

[39] Adriana da Costa F Chaves, Marley M. B. R. Vellasco, and Ricardo Tanscheit. 2005. Fuzzy Rule Extraction from Support Vector Machines. In *5th International Conference on Hybrid Intelligent Systems (HIS 2005), 6-9 November 2005, Rio de Janeiro, Brazil*, Nadia Nedjah, Luiza de Macedo Mourelle, Ajith Abraham, and Mario Köppen (Eds.). IEEE Computer Society, 335–340. https://doi.org/10.1109/ICHIS.2005.51

[40] Zhengping Che, David C. Kale, Wenzhe Li, Mohammad Taha Bahadori, and Yan Liu. 2015. Deep Computational Phenotyping. In *Proceedings of the 21th International Conference on Knowledge Discovery and Data Mining (KDD), Sydney, NSW, Australia, August 10-13, 2015*, Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams (Eds.). ACM, 507–516. https://doi.org/10.1145/2783258.2783365

[41] Fei Chen. 2004. *LEARNING ACCURATE AND UNDERSTANDABLE RULES FROM SVM CLASSIFIERS.* Master's thesis. Simon Fraser University.

[42] Simin Chen, Soroush Bateni, Sampath Grandhi, Xiaodi Li, Cong Liu, and Wei Yang. 2020. DENAS: automated rule generation by knowledge extraction from neural networks. In *ESEC/FSE '20: 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA, November 8-13, 2020*, Prem Devanbu, Myra B. Cohen, and Thomas Zimmermann (Eds.). ACM, 813–825. https://doi.org/10.1145/3368089.3409733

[43] Zhenyu Chen, Jianping Li, and Liwei Wei. 2007. A multiple kernel support vector machine scheme for feature selection and rule extraction from gene expression data of cancer tissue. *Artif. Intell. Medicine* 41, 2 (2007), 161–175. https://doi.org/10.1016/j.artmed.2007.07.008

[44] Kewei Cheng, Ziqing Yang, Ming Zhang, and Yizhou Sun. 2021. UniKER: A Unified Framework for Combining Embedding and Definite Horn Rule Reasoning for Knowledge Graph Inference. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, 9753–9771. https://doi.org/10.18653/v1/2021.emnlp-main.769

[45] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F. Stewart, and Jimeng Sun. 2017. GRAM: Graph-based Attention Model for Healthcare Representation Learning. In *Proceedings of the 23rd ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Halifax, NS, Canada, August 13-17, 2017*. ACM, 787–795. https://doi.org/10.1145/3097983.3098126

[46] Gabriele Ciravegna, Pietro Barbiero, Francesco Giannini, Marco Gori, Pietro Liò, Marco Maggini, and Stefano Melacci. 2023. Logic Explained Networks. *Artif. Intell.* 314 (2023), 103822. https://doi.org/10.1016/j.artint.2022.103822

[47] Peter Clark and Tim Niblett. 1989. The CN2 Induction Algorithm. *Mach. Learn.* 3 (1989), 261–283. https://doi.org/10.1007/BF00116835

[48] William W. Cohen. 1993. Efficient Pruning Methods for Separate-and-Conquer Rule Learning Systems. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence. Chambéry, France, August 28 - September 3, 1993*, Ruzena Bajcsy (Ed.). Morgan Kaufmann, 988–994.

[49] William W. Cohen. 1995. Fast Effective Rule Induction. In *Machine Learning, Proceedings of the Twelfth International Conference on Machine Learning, Tahoe City, California, USA, July 9-12, 1995*, Armand Prieditis and Stuart J. Russell (Eds.). Morgan Kaufmann, 115–123. https://doi.org/10.1016/b978-1-55860-377-6.50023-2

[50] Mark W. Craven and Jude W. Shavlik. 1994. Using Sampling and Queries to Extract Rules from Trained Neural Networks. In *Machine Learning Proceedings 1994*. Elsevier, 37–45. https://doi.org/10.1016/B978-1-55860-335-6.50013-1

[51] Mark W. Craven and Jude W. Shavlik. 1996. Extracting Tree-Structured Representations of Trained Networks. In *Advances in Neural Information Processing Systems 8. Proceedings of the 1995 Conference*, David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo (Eds.). The MIT Press, 24–30. http://papers.nips.cc/paper/1152-extracting-tree-structured-representations-of-trained-networks.pdf

[52] Daniel Cunnington, Mark Law, Jorge Lobo, and Alessandra Russo. 2023. FFNSL: Feed-Forward Neural-Symbolic Learner. *Mach. Learn.* 112, 2 (2023), 515–569. https://doi.org/10.1007/s10994-022-06278-6

[53] Claudia d'Amato, Nicola Flavio Quatraro, and Nicola Fanizzi. 2021. Injecting Background Knowledge into Embedding Models for Predictive Tasks on Knowledge Graphs. In *The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings (Lecture Notes in Computer Science, Vol. 12731)*, Ruben Verborgh, Katja Hose, Heiko Paulheim, Pierre-Antoine Champin, Maria Maleshkova, Óscar Corcho, Petar Ristoski, and Mehwish Alam (Eds.). Springer, 441–457. https://doi.org/10.1007/978-3-030-77385-4_26

[54] Alessandro Daniele and Luciano Serafini. 2019. Knowledge Enhanced Neural Networks. In *PRICAI 2019: Trends in Artificial Intelligence - 16th Pacific Rim International Conference on Artificial Intelligence, Cuvu, Yanuca Island, Fiji, August 26-30, 2019, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 11670)*, Abhaya C. Nayak and Alok Sharma (Eds.). Springer, 542–554. https://doi.org/10.1007/978-3-030-29908-8_43

[55] Tirtharaj Dash, Ashwin Srinivasan, and A. Baskar. 2022. Inclusion of domain-knowledge into GNNs using mode-directed inverse entailment. *Mach. Learn.* 111, 2 (2022), 575–623. https://doi.org/10.1007/s10994-021-06090-8

[56] Tirtharaj Dash, Ashwin Srinivasan, and Lovekesh Vig. 2021. Incorporating symbolic domain knowledge into graph neural networks. *Machine Learning* 110, 7 (2021), 1609–1636. https://doi.org/10.1007/s10994-021-05966-z

[57] Abhinaba Dattachaudhuri, Saroj K. Biswas, Manomita Chakraborty, and Sunita Sarkar. 2021. A transparent rule-based expert system using neural network. *Soft Comput.* 25, 12 (2021), 7731–7744. https://doi.org/10.1007/s00500-020-05547-7

[58] Artur S. d'Avila Garcez and Dov M. Gabbay. 2004. Fibring Neural Networks. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, July 25-29, 2004, San Jose, California, USA*, Deborah L. McGuinness and George Ferguson (Eds.). AAAI Press / The MIT Press, 342–347. http://www.aaai.org/Library/AAAI/2004/aaai04-055.php

[59] Artur S. d'Avila Garcez and Gerson Zaverucha. 1999. The Connectionist Inductive Learning and Logic Programming System. *Appl. Intell.* 11, 1 (1999), 59–77. https://doi.org/10.1023/A:1008328630915

[60] Paulo Vitor de Campos Souza and Edwin Lughofer. 2022. EFNN-NullUni: An evolving fuzzy neural network based on null-uninorm. *Fuzzy Sets Syst.* 449 (2022), 1–31. https://doi.org/10.1016/j.fss.2022.01.010

[61] Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2016. Lifted Rule Injection for Relation Embeddings. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Austin, Texas, USA, November 1-4, 2016*, Jian Su, Xavier Carreras, and Kevin Duh (Eds.). The Association for Computational Linguistics, 1389–1399. https://doi.org/10.18653/v1/d16-1146

[62] Hongyue Diao, Yifan Lu, Ansheng Deng, Li Zou, Xiaofeng Li, and Witold Pedrycz. 2022. Convolutional rule inference network based on belief rule-based system using an evidential reasoning approach. *Knowl. Based Syst.* 237 (2022), 107713. https://doi.org/10.1016/j.knosys.2021.107713

[63] Michelangelo Diligenti, Marco Gori, and Claudio Saccà. 2017. Semantic-Based Regularization for Learning and Inference. *Artificial Intelligence* 244 (2017), 143–165. https://doi.org/10.1016/j.artint.2015.08.011

[64] Michelangelo Diligenti, Soumali Roychowdhury, and Marco Gori. 2017. Integrating Prior Knowledge into Deep Learning. In *Proceedings of the 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, Mexico, December 18-21, 2017*, Xuewen Chen, Bo Luo, Feng Luo, Vasile Palade, and M. Arif Wani (Eds.). IEEE, 920–923. https://doi.org/10.1109/ICMLA.2017.00-37

[65] Honghua Dong, Jiayuan Mao, Tian Lin, Chong Wang, Lihong Li, and Denny Zhou. 2019. Neural Logic Machines. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=B1xY-hRctX

[66] Lu-an Dong, Xin Ye, and Guangfei Yang. 2021. Two-stage rule extraction method based on tree ensemble model for interpretable loan evaluation. *Inf. Sci.* 573 (2021), 46–64. https://doi.org/10.1016/j.ins.2021.05.063

[67] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani (Eds.). ACM, 601–610. https://doi.org/10.1145/2623330.2623623

[68] Mateo Espinosa Zarlenga, Zohreh Shams, and Mateja Jamnik. 2021. Efficient Decompositional Rule Extraction for Deep Neural Networks. *CoRR* abs/2111.12628 (2021). arXiv:2111.12628 https://arxiv.org/abs/2111.12628

[69] Terence A. Etchells and Lisboa Paulo J. G. 2006. Orthogonal search-based rule extraction (OSRE) for trained neural networks: a practical and efficient approach. *IEEE Trans. Neural Networks* 17, 2 (2006), 374–384. https://doi.org/10.1109/TNN.2005.863472

[70] Richard Evans and Edward Grefenstette. 2018. Learning Explanatory Rules from Noisy Data. *Journal of Artificial Intelligence Research* 61 (2018), 1–64. https://doi.org/10.1613/jair.5714

[71] Miao Fan, Qiang Zhou, Emily Chang, and Thomas Fang Zheng. 2014. Transition-based Knowledge Graph Embedding with Relational Mapping Properties. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computation, PACLIC 28, Cape Panwa Hotel, Phuket, Thailand, December 12-14, 2014*, Wirote Aroonmanakun, Prachya Boonkwan, and Thepchai Supnithi (Eds.). The PACLIC 28 Organizing Committee and PACLIC Steering Committee / ACL / Department of Linguistics, Faculty of Arts, Chulalongkorn University, 328–337. https://aclanthology.org/Y14-1039/

[72] Yuan Fang, Kingsley Kuan, Jie Lin, Cheston Tan, and Vijay Chandrasekhar. 2017. Object Detection Meets Knowledge Graphs. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI), Melbourne, Australia, August 19-25, 2017*, Carles Sierra (Ed.). IJCAI, 1661–1667. https://doi.org/10.24963/ijcai.2017/230

[73] Jun Feng, Minlie Huang, Mingdong Wang, Mantong Zhou, Yu Hao, and Xiaoyan Zhu. 2016. Knowledge Graph Embedding by Flexible Translation. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016*, Chitta Baral, James P. Delgrande, and Frank Wolter (Eds.). AAAI Press, 557–560. http://www.aaai.org/ocs/index.php/KR/KR16/paper/view/12887

[74] João Ferreira, Manuel de Sousa Ribeiro, Ricardo Gonçalves, and João Leite. 2022. Looking Inside the Black-Box: Logic-based Explanations for Neural Networks. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning, KR 2022, Haifa, Israel. July 31 - August 5, 2022*, Gabriele Kern-Isberner, Gerhard Lakemeyer, and Thomas Meyer (Eds.). https://proceedings.kr.org/2022/45/

[75] Marc Fischer, Mislav Balunovic, Dana Drachsler-Cohen, Timon Gehr, Ce Zhang, and Martin T. Vechev. 2019. DL2: Training and Querying Neural Networks with Logic. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 1931–1941. http://proceedings.mlr.press/v97/fischer19a.html

[76] Manoel V. M. França, Gerson Zaverucha, and Artur S. d'Avila Garcez. 2014. Fast Relational Learning using Bottom Clause Propositionalization with Artificial Neural Networks. *Machine Learning* 94, 1 (2014), 81–104. https://doi.org/10.1007/s10994-013-5392-1

[77] LiMin Fu. 1994. Rule Generation from Neural Networks. *IEEE Trans. Syst. Man Cybern. Syst.* 24, 8 (1994), 1114–1124. https://doi.org/10.1109/21.299696

[78] Xiuju Fu, ChongJin Ong, S. Keerthi, Gih Guang Hung, and Liping Goh. 2004. Extracting the knowledge embedded in support vector machines. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, Vol. 1. 291–296. https://doi.org/10.1109/IJCNN.2004.1379916

[79] Glenn Fung, Sathyakama Sandilya, and R. Bharat Rao. 2005. Rule extraction from linear support vector machines. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA, August 21-24, 2005*, Robert Grossman, Roberto J. Bayardo, and Kristin P. Bennett (Eds.). ACM, 32–40. https://doi.org/10.1145/1081870.1081878

[80] Johannes Fürnkranz and Gerhard Widmer. 1994. Incremental Reduced Error Pruning. In *Machine Learning, Proceedings of the Eleventh International Conference, Rutgers University, New Brunswick, NJ, USA, July 10-13, 1994*, William W. Cohen and Haym Hirsh (Eds.). Morgan Kaufmann, 70–77. https://doi.org/10.1016/b978-1-55860-335-6.50017-9

[81] Alberto García-Durán, Antoine Bordes, and Nicolas Usunier. 2014. Effective Blending of Two and Three-way Interactions for Modeling Multi-relational Data. In *Proceeding of the 25th Machine Learning and Knowledge Discovery in Databases - European Conference (ECML), Nancy, France, September 15-19, 2014 (Lecture Notes in Computer Science, Vol. 8724)*, Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo (Eds.). Springer, 434–449. https://doi.org/10.1007/978-3-662-44848-9_28

[82] Francesco Giannini, Michelangelo Diligenti, Marco Maggini, Marco Gori, and Giuseppe Marra. 2023. T-norms driven loss functions for machine learning. *Applied Intelligence* (2023), 1–15.

[83] C. Lee Giles and Christian W. Omlin. 1993. Rule refinement with recurrent neural networks. In *Proceedings of International Conference on Neural Networks (ICNN'88), San Francisco, CA, USA, March 28 - April 1, 1993*. IEEE, 801–806. https://doi.org/10.1109/ICNN.1993.298658

[84] Eleonora Giunchiglia and Thomas Lukasiewicz. 2021. Multi-Label Classification Neural Networks with Hard Logical Constraints. *Journal of Artificial Intelligence Research* 72 (2021), 759–818. https://doi.org/10.1613/jair.1.12850

[85] Goran Glavas and Ivan Vulic. 2018. Explicit Retrofitting of Distributional Word Vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Melbourne, Australia, July 15-20, 2018*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, 34–45. https://doi.org/10.18653/v1/P18-1004

[86] Travis R. Goodwin and Dina Demner-Fushman. 2019. Bridging the Knowledge Gap: Enhancing Question Answering with World and Domain Knowledge. *CoRR* abs/1910.07429 (2019). arXiv:1910.07429 http://arxiv.org/abs/1910.07429

[87] Shu Guo, Lin Li, Zhen Hui, Lingshuai Meng, Bingnan Ma, Wei Liu, Lihong Wang, Haibin Zhai, and Hong Zhang. 2020. Knowledge Graph Embedding Preserving Soft Logical Regularity. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 425–434. https://doi.org/10.1145/3340531.3412055

[88] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly Embedding Knowledge Graphs and Logical Rules. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Austin, Texas, USA, November 1-4, 2016*, Jian Su, Xavier Carreras, and Kevin Duh (Eds.). The Association for Computational Linguistics, 192–202. https://doi.org/10.18653/v1/d16-1019

[89] Saman K. Halgamuge and Manfred Glesner. 1994. Neural networks in designing fuzzy systems for real world applications. *Fuzzy Sets and Systems* 65, 1 (1994), 1–12. https://doi.org/10.1016/0165-0114(94)90242-9

[90] Yoichi Hayashi. 1990. A Neural Expert System with Automated Extraction of Fuzzy If-Then Rules. In *Advances in Neural Information Processing Systems 3, [NIPS Conference, Denver, Colorado, USA, November 26-29, 1990]*, Richard Lippmann, John E. Moody, and David S. Touretzky (Eds.). Morgan Kaufmann, 578–584. http://papers.nips.cc/paper/355-a-neural-expert-system-with-automated-extraction-of-fuzzy-if-then-rules

[91] Yoichi Hayashi and Naoki Takano. 2020. One-Dimensional Convolutional Neural Networks with Feature Selection for Highly Concise Rule Extraction from Credit Scoring Datasets with Heterogeneous Attributes. *Electronics* 9, 8 (2020). https://doi.org/10.3390/electronics9081318

[92] Jieyue He, Hae-Jin Hu, R. Harrison, P.C. Tai, and Yi Pan. 2006. Rule generation for protein secondary structure prediction with support vector machines and decision tree. *IEEE Transactions on NanoBioscience* 5, 1 (2006), 46–53. https://doi.org/10.1109/TNB.2005.864021

[93] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. 2015. Learning to Represent Knowledge Graphs with Gaussian Embedding. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis, and Jeffrey Xu Yu (Eds.). ACM, 623–632. https://doi.org/10.1145/2806416.2806502

[94] Tzung-Pei Hong and Jyh-Bin Chen. 1999. Finding relevant attributes and membership functions. *Fuzzy Sets Syst.* 103, 3 (1999), 389–404. https://doi.org/10.1016/S0165-0114(97)00187-5

[95] Tzung-Pei Hong and Jyh-Bin Chen. 2000. Processing individual fuzzy attributes for fuzzy rule induction. *Fuzzy Sets Syst.* 112, 1 (2000), 127–140. https://doi.org/10.1016/S0165-0114(98)00179-1

[96] Tzung-Pei Hong and Chai-Ying Lee. 1996. Induction of fuzzy rules and membership functions from training examples. *Fuzzy Sets Syst.* 84, 1 (1996), 33–47. https://doi.org/10.1016/0165-0114(95)00305-3

[97] Shin-ichi Horikawa, Takeshi Furuhashi, and Yoshiki Uchikawa. 1992. On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm. *IEEE Trans. Neural Networks* 3, 5 (1992), 801–806. https://doi.org/10.1109/72.159069

[98] Vitor A. C. Horta, Ilaria Tiddi, Suzanne Little, and Alessandra Mileo. 2021. Extracting knowledge from Deep Neural Networks through graph analysis. *Future Gener. Comput. Syst.* 120 (2021), 109–118. https://doi.org/10.1016/j.future.2021.02.009

[99] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H. Hovy, and Eric P. Xing. 2016. Harnessing Deep Neural Networks with Logic Rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Berlin, Germany, August 7-12, 2016*. The Association for Computer Linguistics. https://doi.org/10.18653/v1/p16-1228

[100] Zhiting Hu, Zichao Yang, Ruslan Salakhutdinov, and Eric P. Xing. 2016. Deep Neural Networks with Massive Learned Knowledge. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Austin, Texas, USA, November 1-4, 2016*, Jian Su, Xavier Carreras, and Kevin Duh (Eds.). The Association for Computational Linguistics, 1670–1679. https://doi.org/10.18653/v1/d16-1173

[101] Johan Huysmans, Bart Baesens, and Jan Vanthienen. 2006. ITER: An Algorithm for Predictive Regression Rule Extraction. In *Data Warehousing and Knowledge Discovery (DaWaK 2006)*. Springer, 270–279. https://doi.org/10.1007/11823728_26

[102] Muhammad Irfan, Jiangbin Zheng, Muhammad Iqbal, Zafar Masood, and Muhammad Hassan Arif. 2022. Knowledge extraction and retention based continual learning by using convolutional autoencoder-based learning classifier system. *Inf. Sci.* 591 (2022), 287–305. https://doi.org/10.1016/j.ins.2022.01.043

[103] Hisao Ishibuchi, Manabu Nii, and Tadahiko Murata. 1997. Linguistic rule extraction from neural networks and genetic-algorithm-based rule selection. In *Proceedings of International Conference on Neural Networks (ICNN'97), Houston, TX, USA, June 9-12, 1997*. IEEE, 2390–2395. https://doi.org/10.1109/ICNN.1997.614441

[104] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. The Association for Computer Linguistics, 687–696. https://doi.org/10.3115/v1/p15-1067

[105] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge Graph Completion with Adaptive Sparse Transfer Matrix. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, Dale Schuurmans and Michael P. Wellman (Eds.). AAAI Press, 985–991. http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11982

[106] Shichao Jia, Peiwen Lin, Zeyu Li, Jiawan Zhang, and Shixia Liu. 2020. Visualizing surrogate decision trees of convolutional neural networks. *J. Vis.* 23, 1 (2020), 141–156. https://doi.org/10.1007/s12650-019-00607-z

[107] Jingchi Jiang, Huanzheng Wang, Jing Xie, Xitong Guo, Yi Guan, and Qiubin Yu. 2020. Medical knowledge embedding based on recursive neural network for multi-disease diagnosis. *Artificial Intelligence in Medicine* 103 (2020), 101772. https://doi.org/10.1016/j.artmed.2019.101772

[108] Ulf Johansson, Cecilia Sönströd, Tuwe Löfström, and Henrik Boström. 2022. Rule extraction with guarantees from regression models. *Pattern Recognit.* 126 (2022), 108554. https://doi.org/10.1016/j.patcog.2022.108554

[109] DaeEun Kim and Jaeho Lee. 2000. Handling Continuous-Valued Attributes in Decision Tree with Neural Network Modeling. In *Machine Learning: ECML 2000*, Ramon López de Mántaras and Enric Plaza (Eds.). Springer Berlin

Heidelberg, Berlin, Heidelberg, 211–219.

[110] Angelika Kimmig, Stephen Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. 2012. A Short Introduction to Probabilistic Soft Logic. In *Proceedings of the NIPS Workshop on Probabilistic Programming: Foundations and Applications December 07, 2012*. Mansinghka, Vikash, 1–4. $$Uhttps://lirias.kuleuven.be/retrieve/204697$$Dmainarticle[freelyavailable]

[111] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR), Toulon, France, April 24-26, 2017*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl

[112] Rikard Konig, Ulf Johansson, and Lars Niklasson. 2008. G-REX: A Versatile Framework for Evolutionary Data Mining. In *2008 IEEE International Conference on Data Mining Workshops (ICDM 2008 Workshops)*. 971–974. https://doi.org/10.1109/ICDMW.2008.117

[113] R. Krishnan, G. Sivakumar, and P. Bhattacharya. 1999. Extracting decision trees from trained neural networks. *Pattern Recognit.* 32, 12 (1999), 1999–2009. https://doi.org/10.1016/S0031-3203(98)00181-2

[114] R. Krishnan, G. Sivakumar, and P. Bhattacharya. 1999. A search technique for rule extraction from trained neural networks. *Pattern Recognit. Lett.* 20, 3 (1999), 273–280. https://doi.org/10.1016/S0167-8655(98)00145-7

[115] Jens Lehmann, Sebastian Bader, and Pascal Hitzler. 2010. Extracting reduced logic programs from artificial neural networks. *Appl. Intell.* 32, 3 (2010), 249–266. https://doi.org/10.1007/s10489-008-0142-y

[116] Hang-Cheng Li, Kai-Qing Zhou, Li-Ping Mo, Azlan Mohd Zain, and Feng Qin. 2020. Weighted Fuzzy Production Rule Extraction Using Modified Harmony Search Algorithm and BP Neural Network Framework. *IEEE Access* 8 (2020), 186620–186637. https://doi.org/10.1109/ACCESS.2020.3029966

[117] Weidong Li, Rong Peng, and Zhi Li. 2022. Improving knowledge graph completion via increasing embedding interactions. *Appl. Intell.* 52, 8 (2022), 9289–9307. https://doi.org/10.1007/s10489-021-02947-6

[118] Weidong Li, Rong Peng, and Zhi Li. 2023. Knowledge Graph Completion by Jointly Learning Structural Features and Soft Logical Rules. *IEEE Trans. Knowl. Data Eng.* 35, 3 (2023), 2724–2735. https://doi.org/10.1109/TKDE.2021.3108224

[119] Wei Li, Luyao Zhu, Rui Mao, and Erik Cambria. 2023. Skier: A symbolic knowledge integrated model for conversational emotion recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[120] Xiaodan Liang, Zhiting Hu, Hao Zhang, Liang Lin, and Eric P. Xing. 2018. Symbolic Graph Reasoning Meets Convolutions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems (NeurIPS), Montréal, Canada, December 3-8, 2018*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 1858–1868. https://proceedings.neurips.cc/paper/2018/hash/cbb6a3b884f4f88b3a8e3d44c636cbd8-Abstract.html

[121] Qika Lin, Rui Mao, Jun Liu, Fangzhi Xu, and Erik Cambria. 2023. Fusing topology contexts and logical rules in language models for knowledge graph completion. *Inf. Fusion* 90 (2023), 253–264. https://doi.org/10.1016/j.inffus.2022.09.020

[122] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI), Austin, Texas, USA, January 25-30, 2015*, Blai Bonet and Sven Koenig (Eds.). AAAI Press, 2181–2187. http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9571

[123] Bo Liu, Hussein A. Abbass, and Robert I. McKay. 2002. Density-based heuristic for rule discovery with ant-miner. In *The 6th Australia-Japan joint workshop on intelligent and evolutionary system*, Vol. 184.

[124] Bo Liu, Hussein A. Abbass, and Robert I. McKay. 2004. Classification Rule Discovery with Ant Colony Optimization. *IEEE Intell. Informatics Bull.* 3, 1 (2004), 31–35. http://www.comp.hkbu.edu.hk/%7Ecib/2004/Feb/2004/Feb/cib_vol3no1_article4.pdf

[125] Hanxiao Liu, Yuexin Wu, and Yiming Yang. 2017. Analogical Inference for Multi-relational Embeddings. In *Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, NSW, Australia, August 6-11, 2017 (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 2168–2178. http://proceedings.mlr.press/v70/liu17d.html

[126] Quan Liu, Hui Jiang, Zhen-Hua Ling, Si Wei, and Yu Hu. 2016. Probabilistic Reasoning via Deep Learning: Neural Association Models. *CoRR* abs/1603.07704 (2016). arXiv:1603.07704 http://arxiv.org/abs/1603.07704

[127] Tianle Ma and Aidong Zhang. 2018. Multi-view Factorization AutoEncoder with Network Constraints for Multi-omic Integrative Analysis. In *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Madrid, Spain, December 3-6, 2018*, Huiru Jane Zheng, Zoraida Callejas, David Griol, Haiying Wang, Xiaohua Hu, Harald H. H. W. Schmidt, Jan Baumbach, Julie Dickerson, and Le Zhang (Eds.). IEEE Computer Society, 702–707. https://doi.org/10.1109/BIBM.2018.8621379

[128] Matteo Magnini, Giovanni Ciatto, and Andrea Omicini. 2022. KINS: Knowledge Injection via Network Structuring. In *CILC 2022 – Italian Conference on Computational Logic* (Bologna, Italy) *(CEUR Workshop Proceedings, Vol. 3204)*, Roberta Calegari, Giovanni Ciatto, and Andrea Omicini (Eds.). CEUR-WS.org, Bologna, Italy, 254–267. http://ceur-ws.org/Vol-3204/paper_25.pdf

[129] Matteo Magnini, Giovanni Ciatto, and Andrea Omicini. 2022. A view to a KILL: Knowledge Injection via Lambda Layer. In *WOA 2022 – 23rd Workshop "From Objects to Agents"*, Angelo Ferrando and Viviana Mascardi (Eds.). CEUR Workshop Proceedings, Vol. 3261. CEUR-WS.org, Genova, Italy, 61–76. http://ceur-ws.org/Vol-3261/paper5.pdf

[130] Samaneh Mahdavifar and Ali A. Ghorbani. 2020. DeNNeS: deep embedded neural network expert system for detecting cyber attacks. *Neural Comput. Appl.* 32, 18 (2020), 14753–14780. https://doi.org/10.1007/s00521-020-04830-w

[131] Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. 2021. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence* 298 (2021), 103504. https://doi.org/10.1016/j.artint.2021.103504

[132] Emanuele Marconato, Gianpaolo Bontempo, Elisa Ficarra, Simone Calderara, Andrea Passerini, and Stefano Teso. 2023. Neuro Symbolic Continual Learning: Knowledge, Reasoning Shortcuts and Concept Rehearsal. *CoRR* abs/2302.01242 (2023). https://doi.org/10.48550/arXiv.2302.01242 arXiv:2302.01242

[133] Kenneth Marino, Xinlei Chen, Devi Parikh, Abhinav Gupta, and Marcus Rohrbach. 2021. KRISP: Integrating Implicit and Symbolic Knowledge for Open-Domain Knowledge-Based VQA. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 14111–14121. https://doi.org/10.1109/CVPR46437.2021.01389

[134] Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Gupta. 2017. The More You Know: Using Knowledge Graphs for Image Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 20–28. https://doi.org/10.1109/CVPR.2017.10

[135] Urszula Markowska-Kaczmar and Marcin Chumieja. 2004. Discovering the Mysteries of Neural Networks. *Int. J. Hybrid Intell. Syst.* 1, 3-4 (2004), 153–163. http://content.iospress.com/articles/international-journal-of-hybrid-intelligent-systems/his016

[136] Urszula Markowska-Kaczmar and Wojciech Trelak. 2003. Extraction of fuzzy rules from trained neural network using evolutionary algorithm. In *ESANN 2003, 11th European Symposium on Artificial Neural Networks, Bruges, Belgium, April 23-25, 2003, Proceedings*. 149–154. https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2003-9.pdf

[137] Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, and Marco Gori. 2019. LYRICS: a General Interface Layer to Integrate AI and Deep Learning. *CoRR* abs/1903.07534 (2019). arXiv:1903.07534 http://arxiv.org/abs/1903.07534

[138] D. V. Marshakov. 2021. Rule extraction from the Artificial Neural Network. In *IOP Conference Series: Materials Science and Engineering*, Vol. 1029. IOP Publishing, 012127.

[139] David Martens, Bart Baesens, and Tony Van Gestel. 2009. Decompositional Rule Extraction from Support Vector Machines by Active Learning. *IEEE Trans. Knowl. Data Eng.* 21, 2 (2009), 178–191. https://doi.org/10.1109/TKDE.2008.131

[140] David Martens, Manu De Backer, Raf Haesen, Jan Vanthienen, Monique Snoeck, and Bart Baesens. 2007. Classification With Ant Colony Optimization. *IEEE Trans. Evol. Comput.* 11, 5 (2007), 651–665. https://doi.org/10.1109/TEVC.2006.890229

[141] R. Masuoka, N. Watanabe, A. Kawamura, Y. Owada, and K. Asakawa. 1990. Neurofuzzy systems – Fuzzy inference using a structured neural network. In *Proceedings of International Conference on Fuzzy Logic and Neural Networks, Iizuka Japan, July, 1990*. 173–177.

[142] Chirs Matthews and Ilona Jagielska. 1995. Fuzzy rule extraction from a trained multilayer neural network. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, Vol. 2. 744–748 vol.2. https://doi.org/10.1109/ICNN.1995.487510

[143] Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant Supervision for Relation Extraction without Labeled Data. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL) and the 4th International Joint Conference on Natural Language Processing of the AFNLP, Singapore, August 2-7, 2009*, Keh-Yih Su, Jian Su, and Janyce Wiebe (Eds.). The Association for Computer Linguistics (ACL), 1003–1011. https://aclanthology.org/P09-1113/

[144] Sushmita Mitra. 1994. Fuzzy MLP based expert system for medical diagnosis. *Fuzzy Sets and Systems* 65, 2 (1994), 285–296. https://doi.org/10.1016/0165-0114(94)90025-6 Fuzzy Methods for Computer Vision and Pattern Recognition.

[145] Nikola Mrksic, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina Maria Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. 2016. Counter-fitting Word Vectors to Linguistic Constraints. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL), San Diego California, USA, June 12-17, 2016*, Kevin Knight, Ani Nenkova, and Owen Rambow (Eds.). The Association for Computational Linguistics, 142–148. https://doi.org/10.18653/v1/n16-1018

[146] Patrick M. Murphy and Michael J. Pazzani. 1991. ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees. In *Machine Learning Proceedings 1991*. Elsevier, 183–187.

[147] Detlef D. Nauck and Rudolf Kruse. 1997. A neuro-fuzzy method to learn fuzzy classification rules from data. *Fuzzy Sets Syst.* 89, 3 (1997), 277–288. https://doi.org/10.1016/S0165-0114(97)00009-2

[148] Detlef D. Nauck and Rudolf Kruse. 1999. Neuro-fuzzy systems for function approximation. *Fuzzy Sets Syst.* 101, 2 (1999), 261–271. https://doi.org/10.1016/S0165-0114(98)00169-9

[149] Chau Nguyen, Tim French, Wei Liu, and Michael Stewart. 2023. CylE: Cylinder Embeddings for Multi-hop Reasoning over Knowledge Graphs. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, Andreas Vlachos and Isabelle Augenstein (Eds.). Association for Computational Linguistics, 1728–1743. https://aclanthology.org/2023.eacl-main.127

[150] Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. 2016. Holographic Embeddings of Knowledge Graphs. In *Proceedings of the 30th Conference on Artificial Intelligence (AAAI), Phoenix, Arizona, USA, February 12-17, 2016*, Dale Schuurmans and Michael P. Wellman (Eds.). AAAI Press, 1955–1961. http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12484

[151] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of the 28th International Conference on Machine Learning (ICML), Bellevue, Washington, USA, June 28 - July 2, 2011*, Lise Getoor and Tobias Scheffer (Eds.). Omnipress, 809–816. https://icml.cc/2011/papers/438_icmlpaper.pdf

[152] Haydemar Núñez, Cecilio Angulo, and Andreu Català. 2008. Rule Extraction Based on Support and Prototype Vectors. In *Rule Extraction from Support Vector Machines*, Joachim Diederich (Ed.). Studies in Computational Intelligence, Vol. 80. Springer, 109–134. https://doi.org/10.1007/978-3-540-75390-2_5

[153] Josué Obregon and Jae-Yoon Jung. 2023. RuleCOSI+: Rule extraction for interpreting classification tree ensembles. *Inf. Fusion* 89 (2023), 355–381. https://doi.org/10.1016/j.inffus.2022.08.021

[154] Koichi Odajima, Yoichi Hayashi, Gong Tianxia, and Rudy Setiono. 2008. Greedy rule generation from discrete data and its use in neural network rule extraction. *Neural Networks* 21, 7 (2008), 1020–1028. https://doi.org/10.1016/j.neunet.2008.01.003

[155] Simon Odense and Artur S. d'Avila Garcez. 2020. Layerwise Knowledge Extraction from Deep Convolutional Networks. *CoRR* abs/2003.09000 (2020). arXiv:2003.09000 https://arxiv.org/abs/2003.09000

[156] Rafael S. Parpinelli, Heitor S. Lopes, and Alex A. Freitas. 2001. An ant colony based system for data mining: applications to medical data. In *Proceedings of the genetic and evolutionary computation conference (GECCO-2001)*. Citeseer, 791–797.

[157] Baolin Peng, Chunyuan Li, Zhu Zhang, Jinchao Li, Chenguang Zhu, and Jianfeng Gao. 2021. SYNERGY: Building Task Bots at Scale Using Symbolic Knowledge and Machine Teaching. *CoRR* abs/2110.11514 (2021). arXiv:2110.11514 https://arxiv.org/abs/2110.11514

[158] Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 43–54. https://doi.org/10.18653/v1/D19-1005

[159] Gadi Pinkas, Priscila Lima, and Shimon Cohen. 2013. Representing, binding, retrieving and unifying relational knowledge using pools of neural binders. *Biologically Inspired Cognitive Architectures* 6 (2013), 87–95. https://doi.org/10.1016/j.bica.2013.07.005 BICA 2013: Papers from the Fourth Annual Meeting of the BICA Society.

[160] Gadi Pinkus. 1991. Constructing Proofs in Symmetric Networks. In *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2-5, 1991]*, John E. Moody, Stephen Jose Hanson, and Richard Lippmann (Eds.). Morgan Kaufmann, 217–224. http://papers.nips.cc/paper/473-constructing-proofs-in-symmetric-networks

[161] Emanoil Pop, Ross Hayward, and Joachim Diederich. 1994. *RULENEG: Extracting rules from a trained ANN by stepwise negation.* Technical Report. Neurocomputing Research Centre, Queensland University of Technology.

[162] Mohsen Pourvali, Yao Meng, Chen Sheng, and Yangzhou Du. 2023. TaxoKnow: Taxonomy as Prior Knowledge in the Loss Function of Multi-class Classification. *CoRR* abs/2305.16341 (2023). https://doi.org/10.48550/arXiv.2305.16341 arXiv:2305.16341

[163] J. Ross Quinlan. 1986. Induction of Decision Trees. *Mach. Learn.* 1, 1 (1986), 81–106. https://doi.org/10.1023/A:1022643204877

[164] J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kauffmann, San Mateo, CA, USA. https://dl.acm.org/doi/10.5555/152181

[165] Juan R. Rabuñal, Julian Dorado, Alejandro Pazos, Javier Pereira, and Daniel Rivero. 2004. A New Approach to the Extraction of ANN Rules and to Their Generalization Capacity Through GP. *Neural Comput.* 16, 7 (2004), 1483–1523. https://doi.org/10.1162/089976604323057461

[166] Hongyu Ren and Jure Leskovec. 2020. Beta Embeddings for Multi-Hop Logical Reasoning in Knowledge Graphs. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/e43739bba7cdb577e9e3e4e42447f5a5-

Abstract.html

[167] Ryan Riegel, Alexander G. Gray, Francois P. S. Luus, Naweed Khan, Ndivhuwo Makondo, Ismail Yunus Akhalwaya, Haifeng Qian, Ronald Fagin, Francisco Barahona, Udit Sharma, Shajith Ikbal, Hima Karanam, Sumit Neelam, Ankita Likhyani, and Santosh K. Srivastava. 2020. Logical Neural Networks. *CoRR* abs/2006.13155 (2020). arXiv:2006.13155 https://arxiv.org/abs/2006.13155

[168] Tim Rocktäschel and Sebastian Riedel. 2017. End-to-end Differentiable Proving. In *Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems, Long Beach, CA, USA, December 4-9, 2017*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 3788–3800. https://proceedings.neurips.cc/paper/2017/hash/b2ab001909a8a6f04b51920306046ce5-Abstract.html

[169] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. 2015. Injecting Logical Background Knowledge into Embeddings for Relation Extraction. In *The 2015 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies (HLT), Denver, Colorado, USA, May 31 - June 5, 2015*, Rada Mihalcea, Joyce Yue Chai, and Anoop Sarkar (Eds.). The Association for Computational Linguistics, 1119–1129. https://doi.org/10.3115/v1/n15-1118

[170] Natalia Díaz Rodríguez, Alberto Lamas, Jules Sanchez, Gianni Franchi, Ivan Donadello, Siham Tabik, David Filliat, Policarpo Cruz, Rosana Montes, and Francisco Herrera. 2022. EXplainable Neural-Symbolic Learning (*X-NeSyL*) methodology to fuse deep learning representations with expert knowledge graphs: The MonuMAI cultural heritage use case. *Inf. Fusion* 79 (2022), 58–83. https://doi.org/10.1016/j.inffus.2021.09.022

[171] Emad W. Saad and Donald C. Wunsch II. 2007. Neural network explanation using inversion. *Neural Networks* 20, 1 (2007), 78–93. https://doi.org/10.1016/j.neunet.2006.07.005

[172] Federico Sabbatini and Roberta Calegari. 2022. Symbolic Knowledge Extraction from Opaque Machine Learning Predictors: GridREx & PEDRO. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning, KR 2022, Haifa, Israel. July 31 - August 5, 2022*, Gabriele Kern-Isberner, Gerhard Lakemeyer, and Thomas Meyer (Eds.). https://proceedings.kr.org/2022/57/

[173] Federico Sabbatini and Roberta Calegari. 2023. Explainable Black Boxes via Explainable Clustering: ExACT and CReEPy. *Decision Support Systems (submitted to)* (2023).

[174] Federico Sabbatini, Giovanni Ciatto, and Andrea Omicini. 2021. GridEx: An Algorithm for Knowledge Extraction from Black-Box Regressors. In *Explainable and Transparent AI and Multi-Agent Systems. Third International Workshop, EXTRAAMAS 2021, Virtual Event, May 3–7, 2021, Revised Selected Papers*, Davide Calvaresi, Amro Najjar, Michael Winikoff, and Kary Främling (Eds.). LNCS, Vol. 12688. Springer Nature, Basel, Switzerland, 18–38. https://doi.org/10.1007/978-3-030-82017-6_2

[175] Kazumi Saito and Ryohei Nakano. 1988. Medical diagnostic expert system based on PDP model. In *Proceedings of International Conference on Neural Networks (ICNN'88), San Diego, CA, USA, July 24-27, 1988*. IEEE, 255–262. https://doi.org/10.1109/ICNN.1988.23855

[176] Kazumi Saito and Ryohei Nakano. 1997. Law Discovery using Neural Networks. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, IJCAI 97, Nagoya, Japan, August 23-29, 1997, 2 Volumes*. Morgan Kaufmann, 1078–1083. http://ijcai.org/Proceedings/97-2/Papers/042.pdf

[177] Kazumi Saito and Ryohei Nakano. 2002. Extracting regression rules from neural networks. *Neural Networks* 15, 10 (2002), 1279–1288. https://doi.org/10.1016/S0893-6080(02)00089-8

[178] Armin Salimi-Badr and Mohammad Mehdi Ebadzadeh. 2022. A novel learning algorithm based on computing the rules' desired outputs of a TSK fuzzy neural network with non-separable fuzzy rules. *Neurocomputing* 470 (2022), 139–153. https://doi.org/10.1016/j.neucom.2021.10.103

[179] Makoto Sato and Hiroshi Tsukimoto. 2001. Rule extraction from neural networks via decision tree induction. In *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)*, Vol. 3. IEEE, 1870–1875. https://doi.org/10.1109/IJCNN.2001.938448

[180] Vitaly Schetinin, Jonathan E. Fieldsend, Derek Partridge, Timothy J. Coats, Wojtek J. Krzanowski, Richard M. Everson, Trevor C. Bailey, and Adolfo Hernandez. 2007. Confident Interpretation of Bayesian Decision Tree Ensembles for Clinical Applications. *IEEE Trans. Inf. Technol. Biomed.* 11, 3 (2007), 312–319. https://doi.org/10.1109/TITB.2006.880553

[181] Gregor P. J. Schmitz, Chris Aldrich, and François S. Gouws. 1999. ANN-DT: an algorithm for extraction of decision trees from artificial neural networks. *IEEE Transactions on Neural Networks* 10, 6 (1999), 1392–1401. https://doi.org/10.1109/72.809084

[182] Prithviraj Sen, Breno W. S. R. de Carvalho, Ryan Riegel, and Alexander G. Gray. 2022. Neuro-Symbolic Inductive Logic Programming with Logical Neural Networks. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 8212–8219. https://ojs.aaai.org/index.php/AAAI/article/view/20795

[183] Sabrina Sestito and Tharam S. Dillon. 1994. *Automated knowledge acquisition.* Prentice Hall.

[184] Kamal Kumar Sethi, Durgesh Kumar Mishra, and Bharat Mishra. 2012. KDRuleEx: A Novel Approach for Enhancing User Comprehensibility Using Rule Extraction. In *2012 Third International Conference on Intelligent Systems Modelling and Simulation*. 55–60. https://doi.org/10.1109/ISMS.2012.116

[185] Rudy Setiono. 1997. Extracting Rules from Neural Networks by Pruning and Hidden-Unit Splitting. *Neural Comput.* 9, 1 (1997), 205–225. https://doi.org/10.1162/neco.1997.9.1.205

[186] Rudy Setiono. 2000. Extracting M-of-N rules from trained neural networks. *IEEE Trans. Neural Networks Learn. Syst.* 11, 2 (2000), 512–519. https://doi.org/10.1109/72.839020

[187] Rudy Setiono, Bart Baesens, and Christophe Mues. 2008. Recursive Neural Network Rule Extraction for Data With Mixed Attributes. *IEEE Trans. Neural Networks* 19, 2 (2008), 299–307. https://doi.org/10.1109/TNN.2007.908641

[188] Rudy Setiono and Wee Kheng Leow. 2000. FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks. *Appl. Intell.* 12, 1-2 (2000), 15–25. https://doi.org/10.1023/A:1008307919726

[189] Rudy Setiono, Wee Kheng Leow, and Jacek M. Zurada. 2002. Extraction of rules from artificial neural networks for nonlinear regression. *IEEE Transactions on Neural Networks* 13, 3 (2002), 564–577. https://doi.org/10.1109/TNN.2002.1000125

[190] Rudy Setiono and Huan Liu. 1996. Symbolic Representation of Neural Networks. *Computer* 29, 3 (1996), 71–77. https://doi.org/10.1109/2.485895

[191] Rudy Setiono and Huan Liu. 1997. NeuroLinear: A System for Extracting Oblique Decision Rules from Neural Networks. In *Machine Learning: ECML-97, 9th European Conference on Machine Learning, Prague, Czech Republic, April 23-25, 1997, Proceedings (Lecture Notes in Computer Science, Vol. 1224)*, Maarten van Someren and Gerhard Widmer (Eds.). Springer, 221–233. https://doi.org/10.1007/3-540-62858-4_87

[192] Rudy Setiono and James Y. L. Thong. 2004. An approach to generate rules from neural networks for regression problems. *Eur. J. Oper. Res.* 155, 1 (2004), 239–250. https://doi.org/10.1016/S0377-2217(02)00792-0

[193] Zohreh Shams, Botty Dimanov, Sumaiyah Kola, Nikola Simidjievski, Helena Andres Terre, Paul Scherer, Urška Matjašec, Jean Abraham, Pietro Liò, and Mateja Jamnik. 2021. REM: An integrative rule extraction methodology for explainable data analysis in healthcare. *medRxiv* (2021), 2021–01.

[194] Alisa Smirnova, Jie Yang, Dingqi Yang, and Philippe Cudre-Mauroux. 2022. Nessy: A Neuro-Symbolic System for Label Noise Reduction. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[195] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 926–934. https://proceedings.neurips.cc/paper/2013/hash/b337e84de8752b27eda3a12363109e80-Abstract.html

[196] Gustav Sourek, Vojtech Aschenbrenner, Filip Zelezný, Steven Schockaert, and Ondrej Kuzelka. 2018. Lifted Relational Neural Networks: Efficient Learning of Latent Relational Structures. *Journal of Artificial Intelligence Research* 62 (2018), 69–100. https://doi.org/10.1613/jair.1.11203

[197] Giuseppe Spillo, Cataldo Musto, Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. 2022. Knowledge-aware Recommendations Based on Neuro-Symbolic Graph Embeddings and First-Order Logical Rules. In *RecSys '22: Sixteenth ACM Conference on Recommender Systems, Seattle, WA, USA, September 18 - 23, 2022*, Jennifer Golbeck, F. Maxwell Harper, Vanessa Murdock, Michael D. Ekstrand, Bracha Shapira, Justin Basilico, Keld T. Lundgaard, and Even Oldridge (Eds.). ACM, 616–621. https://doi.org/10.1145/3523227.3551484

[198] Russell Stewart and Stefano Ermon. 2017. Label-Free Supervision of Neural Networks with Physics and Domain Knowledge. In *Proceedings of the 31st Conference on Artificial Intelligence (AAAI), San Francisco, California, USA, February 4-9, 2017*, Satinder P. Singh and Shaul Markovitch (Eds.). AAAI Press, 2576–2582. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14967

[199] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *Proceedings of the 7th International Conference on Learning Representations (ICLR), New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=HkgEQnRqYQ

[200] Ismail A. Taha and Joydeep Ghosh. 1999. Symbolic Interpretation of Artificial Neural Networks. *IEEE Trans. Knowl. Data Eng.* 11, 3 (1999), 448–463. https://doi.org/10.1109/69.774103

[201] Ah-Hwee Tan. 1997. Cascade ARTMAP: Integrating Neural Computation and Symbolic Knowledge Processing. *IEEE Transaction on Neural Networks* 8, 2 (1997), 237–250. https://doi.org/10.1109/72.557661

[202] Xiaojuan Tang, Song-Chun Zhu, Yitao Liang, and Muhan Zhang. 2022. RulE: Neural-Symbolic Knowledge Graph Reasoning with Rule Embedding. *CoRR* abs/2210.14905 (2022). https://doi.org/10.48550/arXiv.2210.14905 arXiv:2210.14905

[203] Sebastian B. Thrun. 1993. *Extracting Provably Correct Rules from Artificial Neural Networks.* Technical Report. University of Bonn.

[204] Alan B. Tickle, Marian Orlowski, and Joachim Diederich. 1996. DEDEC: A methodology for extracting rules from trained artificial neural networks. In *Rules and Networks: Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop*, Robert Andrews and Joachim Diederich (Eds.). Neurocomputing Research Centre, Queensland University of Technology, 90–102.

[205] Douglas E. D. Torres and Claudio M. S. Rocco. 2005. Extracting Trees from Trained SVM Models using a TREPAN Based Approach. In *5th International Conference on Hybrid Intelligent Systems (HIS 2005), 6-9 November 2005, Rio de Janeiro, Brazil*, Nadia Nedjah, Luiza de Macedo Mourelle, Ajith Abraham, and Mario Köppen (Eds.). IEEE Computer Society, 353–360. https://doi.org/10.1109/ICHIS.2005.41

[206] Geoffrey G. Towell and Jude W. Shavlik. 1991. Interpretation of Artificial Neural Networks: Mapping Knowledge-Based Neural Networks into Rules. In *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2-5, 1991]*, John E. Moody, Stephen Jose Hanson, and Richard Lippmann (Eds.). Morgan Kaufmann, 977–984. http://papers.nips.cc/paper/546-interpretation-of-artificial-neural-networks-mapping-knowledge-based-neural-networks-into-rules

[207] Geoffrey G. Towell and Jude W. Shavlik. 1993. Extracting refined rules from knowledge-based neural networks. *Machine Learning* 13, 1 (1993), 71–101. https://doi.org/10.1007/BF00993103

[208] Geoffrey G. Towell, Jude W. Shavlik, and Michiel O. Noordewier. 1990. Refinement ofApproximate Domain Theories by Knowledge-Based Neural Networks. In *Proceedings of the 8th National Conference on Artificial Intelligence. Boston, Massachusetts, USA, July 29 - August 3, 1990, 2 Volumes*, Howard E. Shrobe, Thomas G. Dietterich, and William R. Swartout (Eds.). AAAI Press / The MIT Press, 861–866. http://www.aaai.org/Library/AAAI/1990/aaai90-129.php

[209] Son N. Tran and Artur S. d'Avila Garcez. 2016. Deep Logic Networks: Inserting and Extracting Knowledge From Deep Belief Networks. *IEEE Transaction on Neural Networks and Learning Systems* 29, 2 (2016), 246–258. https://doi.org/10.1109/TNNLS.2016.2603784

[210] Volker Tresp, Jürgen Hollatz, and Subutai Ahmad. 1992. Network Structuring and Training Using Rule-Based Knowledge. In *Advances in Neural Information Processing Systems 5, [NIPS Conference, Denver, Colorado, USA, November 30 - December 3, 1992]*, Stephen Jose Hanson, Jack D. Cowan, and C. Lee Giles (Eds.). Morgan Kaufmann, 871–878. http://papers.nips.cc/paper/638-network-structuring-and-training-using-rule-based-knowledge

[211] Volker Tresp, Jürgen Hollatz, and Subutai Ahmad. 1992. Network Structuring and Training Using Rule-Based Knowledge. In *Advances in Neural Information Processing Systems 5, [NIPS Conference, Denver, Colorado, USA, November 30 - December 3, 1992]*, Stephen Jose Hanson, Jack D. Cowan, and C. Lee Giles (Eds.). Morgan Kaufmann, 871–878. http://papers.nips.cc/paper/638-network-structuring-and-training-using-rule-based-knowledge

[212] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33rd International Conference on Machine Learning (ICML), New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings, Vol. 48)*, Maria-Florina Balcan and Kilian Q. Weinberger (Eds.). JMLR.org, 2071–2080. http://proceedings.mlr.press/v48/trouillon16.html

[213] Hiroshi Tsukimoto. 2000. Extracting rules from trained neural networks. *IEEE Trans. Neural Networks Learn. Syst.* 11, 2 (2000), 377–389. https://doi.org/10.1109/72.839008

[214] Nikola Vasilev, Zheni Mincheva, and Ventsislav Nikolov. 2020. Decision Tree Extraction using Trained Neural Network. In *Proceedings of the 9th International Conference on Smart Cities and Green ICT Systems, SMARTGREENS 2020, Prague, Czech Republic, May 2-4, 2020*, Cornel Klein and Markus Helfert (Eds.). SCITEPRESS, 194–200. https://doi.org/10.5220/0009351801940200

[215] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *Proceedings of the 6th International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, April 30 - May 3, 2018*. OpenReview.net. https://openreview.net/forum?id=rJXMpikCZ

[216] Quan Wang, Bin Wang, and Li Guo. 2015. Knowledge Base Completion Using Embeddings and Rules. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI), Buenos Aires, Argentina, July 25-31, 2015*, Qiang Yang and Michael J. Wooldridge (Eds.). AAAI Press, 1859–1866. http://ijcai.org/Abstract/15/264

[217] Sutong Wang, Yuyan Wang, Dujuan Wang, Yunqiang Yin, Yanzhang Wang, and Yaochu Jin. 2020. An improved random forest-based rule extraction method for breast cancer diagnosis. *Appl. Soft Comput.* 86 (2020). https://doi.org/10.1016/j.asoc.2019.105941

[218] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *Proceedings of the 28th Conference on Artificial Intelligence (AAAI), Québec City, Québec, Canada, July 27 -31, 2014*, Carla E. Brodley and Peter Stone (Eds.). AAAI Press, 1112–1119. http://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8531

[219] Zhepei Wei, Yue Wang, Jinnan Li, Zhining Liu, Erxin Yu, Yuan Tian, Xin Wang, and Yi Chang. 2022. Towards Unified Representations of Knowledge Graph and Expert Rules for Machine Learning and Reasoning. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2022 - Volume 1: Long Papers, Online Only, November*

20-23, 2022. Association for Computational Linguistics, 240–253. https://aclanthology.org/2022.aacl-main.20

[220] Zhuoyu Wei, Jun Zhao, Kang Liu, Zhenyu Qi, Zhengya Sun, and Guanhua Tian. 2015. Large-scale Knowledge Base Completion: Inferring via Grounding Network Sampling over Selected Instances. In *Proceedings of the 24th International Conference on Information and Knowledge Management (CIKM), Melbourne, VIC, Australia, October 19 - 23, 2015*, James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis, and Jeffrey Xu Yu (Eds.). ACM, 1331–1340. https://doi.org/10.1145/2806416.2806513

[221] Luisa Werner, Nabil Layaïda, Pierre Genevès, and Sarah Chlyah. 2023. Knowledge Enhanced Graph Neural Networks. *CoRR* abs/2303.15487 (2023). https://doi.org/10.48550/arXiv.2303.15487 arXiv:2303.15487

[222] Peter West, Chandra Bhagavatula, Jack Hessel, Jena D. Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic Knowledge Distillation: from General Language Models to Commonsense Models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, Marine Carpuat, Marie-Catherine de Marneffe, and Iván Vladimir Meza Ruíz (Eds.). Association for Computational Linguistics, 4602–4625. https://doi.org/10.18653/v1/2022.naacl-main.341

[223] Han Xiao, Minlie Huang, Yu Hao, and Xiaoyan Zhu. 2015. TransA: An Adaptive Approach for Knowledge Graph Embedding. *CoRR* abs/1509.05490 (2015). arXiv:1509.05490 http://arxiv.org/abs/1509.05490

[224] Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. From One Point to a Manifold: Knowledge Graph Embedding for Precise Link Prediction. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, Subbarao Kambhampati (Ed.). IJCAI/AAAI Press, 1315–1321. http://www.ijcai.org/Abstract/16/190

[225] Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. TransG : A Generative Model for Knowledge Graph Embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics. https://doi.org/10.18653/v1/p16-1219

[226] Yaqi Xie, Fan Zhou, and Harold Soh. 2021. Embedding Symbolic Temporal Knowledge into Deep Sequential Models. In *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*. IEEE, 4267–4273. https://doi.org/10.1109/ICRA48506.2021.9561952

[227] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. 2018. A Semantic Loss Function for Deep Learning with Symbolic Knowledge. In *Proceedings of the 35th International Conference on Machine Learning (ICML), Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer G. Dy and Andreas Krause (Eds.). PMLR, 5498–5507. http://proceedings.mlr.press/v80/xu18h.html

[228] Anli Yan, Zhenxiang Chen, Haibo Zhang, Lizhi Peng, Qiben Yan, Muhammad Umair Hassan, Chuan Zhao, and Bo Yang. 2021. Effective detection of mobile malware behavior based on explainable deep neural network. *Neurocomputing* 453 (2021), 482–492. https://doi.org/10.1016/j.neucom.2020.09.082

[229] Bishan Yang, Wen-Tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR), San Diego, CA, USA, May 7-9, 2015*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6575

[230] Cheng Yang, Jiawei Liu, and Chuan Shi. 2021. Extract the Knowledge of Graph Neural Networks and Go Beyond it: An Effective Knowledge Distillation Framework. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 1227–1237. https://doi.org/10.1145/3442381.3450068

[231] Dounia Yedjour. 2021. Application of the Genetic Algorithm to the Rule Extraction Problem. In *Artificial Intelligence and Renewables Towards an Energy Transition*, Mustapha Hatti (Ed.). Springer International Publishing, Cham, 604–611.

[232] Dounia Yedjour and Abdelkader Benyettou. 2018. Symbolic interpretation of artificial neural networks based on multiobjective genetic algorithms and association rules mining. *Appl. Soft Comput.* 72 (2018), 177–188. https://doi.org/10.1016/j.asoc.2018.08.007

[233] Dongran Yu, Bo Yang, Qianhao Wei, Anchen Li, and Shirui Pan. 2022. A Probabilistic Graphical Model Based on Neural-symbolic Reasoning for Visual Relationship Detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 10599–10608. https://doi.org/10.1109/CVPR52688.2022.01035

[234] Jianbo Yu and Guoliang Liu. 2021. Extracting and Inserting Knowledge into Stacked Denoising Auto-Encoders. *Neural Networks* 137 (2021), 31–42. https://doi.org/10.1016/j.neunet.2021.01.010

[235] Yufei Yuan and Huijun Zhuang. 1996. A genetic algorithm for generating fuzzy classification rules. *Fuzzy Sets Syst.* 84, 1 (1996), 1–19. https://doi.org/10.1016/0165-0114(95)00302-9

[236] Ying Zhang, Hongye Su, Tao Jia, and Jian Chu. 2005. Rule Extraction from Trained Support Vector Machines. In *Advances in Knowledge Discovery and Data Mining, 9th Pacific-Asia Conference, PAKDD 2005, Hanoi, Vietnam, May*

18-20, 2005, Proceedings (Lecture Notes in Computer Science, Vol. 3518), Tu Bao Ho, David Wai-Lok Cheung, and Huan Liu (Eds.). Springer, 61–70. https://doi.org/10.1007/11430919_9

[237] Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction. In *Proceedings of the 34th Conference on Artificial Intelligence (AAAI), The 32nd Innovative Applications of Artificial Intelligence Conference (IAAI), The 10th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI), New York, NY, USA, February 7-12, 2020*. AAAI Press, 3065–3072. https://aaai.org/ojs/index.php/AAAI/article/view/5701

[238] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL), Florence, Italy, July 28 - August 2, 2019*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, 1441–1451. https://doi.org/10.18653/v1/p19-1139

[239] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense Knowledge Aware Conversation Generation with Graph Attention. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI), Stockholm, Sweden, July 13-19, 2018*, Jérôme Lang (Ed.). ijcai.org, 4623–4629. https://doi.org/10.24963/ijcai.2018/643

[240] Zhi-Hua Zhou, Shi-Fu Chen, and Zhaoqian Chen. 2000. A Statistics Based Approach for Extracting Priority Rules from Trained Neural Networks. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000, Neural Computing: New Challenges and Perspectives for the New Millennium, Como, Italy, July 24-27, 2000, Volume 3*. IEEE Computer Society, 401–406. https://doi.org/10.1109/IJCNN.2000.861337

[241] Zhi-Hua Zhou, Yuan Jiang, and Shi-Fu Chen. 2003. Extracting symbolic rules from trained neural network ensembles. *AI Commun.* 16, 1 (2003), 3–15. http://content.iospress.com/articles/ai-communications/aic272

[242] Zhaocheng Zhu, Mikhail Galkin, Zuobai Zhang, and Jian Tang. 2022. Neural-Symbolic Models for Logical Queries on Knowledge Graphs. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (Eds.). PMLR, 27454–27478. https://proceedings.mlr.press/v162/zhu22c.html

[243] Jan Ruben Zilke, Eneldo Loza Mencía, and Frederik Janssen. 2016. DeepRED – Rule Extraction from Deep Neural Networks. In *Discovery Science (Lecture Notes in Computer Science, Vol. 9956)*, Toon Calders, Michelangelo Ceci, and Donato Malerba (Eds.). Springer, Bari, Italy, 457–473. https://doi.org/10.1007/978-3-319-46307-0_29