

Heuristic Algorithm for Conceptual Learning [†]

Dimitar Minkov ^{1,*} , Roumen Trifonov ¹  and Dilyana Budakova ^{2,*} 

¹ Faculty of Computer Systems and Technologies, Technical University of Sofia, 1000 Sofia, Bulgaria; r_trifonov@tu-sofia.bg

² Computer Systems and Technology Department, Technical University of Sofia, Branch Plovdiv, 4000 Plovdiv, Bulgaria

* Correspondence: dimitar_minkoff@abv.bg (D.M.); dilyana_budakova@yahoo.com (D.B.)

[†] Presented at the 15th International Scientific Conference TechSys 2026—Engineering, Technologies and Systems, Plovdiv, Bulgaria, 14–16 May 2026.

Abstract

This article presents a new heuristic algorithm for conceptual learning. The process of constructing a decision tree through the minimization of a logical function is explained. This heuristic approach eliminates the need to expand the inverse property space into a full logical expression, which could be computationally expensive for large expressions. Tests were conducted using various sets of standardized machine learning benchmarks. They show that the algorithm operates with high precision and accuracy on noise-free data as well as on datasets that include noise through the presence of mislabeled examples. The algorithm's ability to capture non-linear patterns was experimentally evaluated, where the target class is defined by the equality of two specified attributes. The application of the proposed algorithm for the creation of hybrid neuro-symbolic architectures is discussed, with the aim of achieving logically grounded and interpretable machine learning and machine reasoning.

Keywords: concept learning; decision tree; logical function minimization; heuristic approach

1. Introduction

Scientific research in the field of artificial intelligence demonstrates achievements in the creation and application of cognitive models of reasoning, connectionist models for deep learning, transformer models, and large language models. Recently, increasing efforts have been directed toward integrating, within a single cognitive behavioral model, the achievements of the connectionist approach in machine learning and automated symbolic reasoning. This will lead to the development of increasingly reliable and more interpretable neuro-symbolic systems for learning and reasoning.

In conceptual learning, the goal is to learn a given concept from a set of positive and negative examples that describe it [1]. During the learning process, a hypothesis is sought that correctly classifies the examples, i.e., it corresponds to all positive examples and does not correspond to the negative ones. This hypothesis represents the learned notion (concept). Examples of conceptual learning include supervised learning algorithms such as decision trees, version space learning, explanation-based learning (EBL), and few-shot learning, which directly aim to learn concepts from examples.

According to [2], informed machine learning requires a hybrid information source consisting of data and prior knowledge. An independent source provides the prior knowledge. It is represented formally and explicitly integrated at different stages of the machine



Academic Editors: Nikola G. Shakev,
Valyo N. Nikolov, Nikolay R.
Kakanakov and Sevil A.
Ahmed-Shieva

Published: 22 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

learning process and in various ways. An overview of techniques for integrating prior knowledge into the machine learning process is presented in [2]. Tools for integrated machine learning and reasoning are presented in [3]. The introduction of these hybrid models aims, for example, to enable machine learning to be performed with less data, to improve its performance, and to ensure that the acquired knowledge is reliable, i.e., to avoid hallucinations.

The prior domain knowledge obtained through conceptual learning is represented in symbolic form, such as logic rules or logical constraints, attributes, feature vectors, concepts, causal relationships, relations in first-order logic, probability distributions, knowledge graphs, decision trees, and others. In [4], an overview is provided of the inclusion of domain knowledge in deep neural networks by means of changes to the input, the loss function, and the architecture of deep networks. Knowledge about a given domain can also be incorporated through dialogue thanks to large language models and the attention mechanism [5]. The benefit of incorporating domain knowledge is the creation of models with better predictive performance [6]. The inclusion of domain knowledge also plays an important role in generating explanations of how decisions are made in models with deep neural networks, which will lead to increased trust in artificial intelligence systems and the achievement of Explainable Artificial Intelligence (EAI) [4]. One solution is the implementation of chain-of-thought agents, which involve layering a reasoning controller on top of the base Large Language Model (LLM) [7]. Another solution is the development of decoding methods that directly guide a Pre-Trained Language Model toward target keywords. In [8], a two-system decoding framework, Decider, is described—where the base LLMs are equipped with a First-Order Logic (FOL) reasoner along with a decision function that merges the outputs to guide the generation.

Therefore, it is important to develop new and more reliable algorithms and heuristics for machine learning and reasoning. In this article, a new heuristic algorithm for rule extraction from examples is proposed. The decision tree is constructed using a heuristic that involves the minimization of logical functions. A similar approach to conceptual learning is presented in [9]. In the algorithm proposed herein, the following new heuristic is introduced: from the shortest representation of the property space, covering rules are extracted directly by considering only one property from each row. In order to determine the most useful property, it is only necessary to count in how many examples of the corresponding class the given value of that property occurs. In this way, the need to expand the logical expression is avoided, thereby saving computational resources.

The paper is structured as follows: Section 2 presents relevant related work. Section 3 presents in detail the proposed heuristic for the minimization of logical functions in order to construct a decision tree. Section 4 describes the implementation of the proposed method. Section 5 presents and analyzes the experiments and experimental results, and Section 6 provides conclusions and outlines future trends for intelligent system modeling.

2. Related Work

Decision tree learning and rule induction are well-established approaches for classification and knowledge extraction. Classical algorithms such as CART [10] and C4.5 [11] employ greedy, top-down strategies that optimize local splitting criteria without guaranteeing global optimality. In parallel, rule induction methods such as CN2 [12] and RIPPER [13] construct interpretable rule sets using a separate-and-conquer paradigm.

Subsequent work has explored both improved optimality and interpretability. For example, ref. [14] considers complete search strategies for decision trees, while in [15], domain-adaptive trees with implications for fairness are studied. Methods such as TREPAN [16] and more recent rule-based learning approaches [17] further emphasize extracting compact,

interpretable representations from data or trained models. Within this landscape, the proposed method can be viewed as an alternative heuristic that aims to balance efficiency and compact rule extraction, consistent with ongoing work in Explainable Artificial Intelligence.

3. Logical Function Minimization for Building Decision Trees

In the context of knowledge extraction, the input data are typically represented as objects within a defined feature space. Each object is characterized by a set of attributes and an associated class label. The aim is to construct a model that formalizes the relationship between objects in the feature space and the set of admissible classes, ensuring consistency with the observed examples. Such a model enables the classification of previously unseen objects based solely on their attribute values.

This formulation is closely related to the problem of minimizing a logical function with partially specified input configurations. A subset of input assignments is known, together with corresponding function values equal to 0 or 1. The objective is to determine a fully specified function that produces an output for every possible input configuration, preserves agreement with the known assignments, and is expressed in a representation of minimal complexity.

The proposed method considers the known examples, defined by their properties, as the arrangements of the arguments for which the function is defined. When simplifying a logical function, an equivalent function is found that corresponds to the defined arrangements. The objective of this procedure is to obtain a representation of minimal length.

Within the context of learning from examples, this corresponds to constructing a model of the property space that captures the known examples. The goal is to ensure that the resulting model remains as concise as possible while preserving correctness with respect to the observed data.

Assume that the data describing the known examples within the property space is organized in tabular form, where each column corresponds to a distinct property and each row contains the specific property values associated with a particular example. Such a table may be interpreted as a logical function expressed in disjunctive normal form (DNF). In this representation, each row corresponds to a conjunction of property values that characterizes a single example, while the collection of rows is combined through disjunction, forming the overall logical expression.

By applying a logical function minimization method, the resulting function covers all known positive configurations of the arguments, as well as a subset of configurations that were initially undefined. The remaining configurations—including both explicitly negative and undefined cases not covered by the function—are treated as negative. Analogously, in learning from examples, a model can be constructed that encompasses all positive examples and possibly some previously unknown instances, while all remaining instances are classified as negative. The proposed method implements this strategy.

4. Description of the Method Implementation

4.1. Preprocessing

Figure 1 illustrates the implementation flowchart. To evaluate the effectiveness of the proposed method, the dataset is randomly partitioned into two subsets: two-thirds of the samples are used to construct the model tree, while the remaining one-third is reserved for testing.

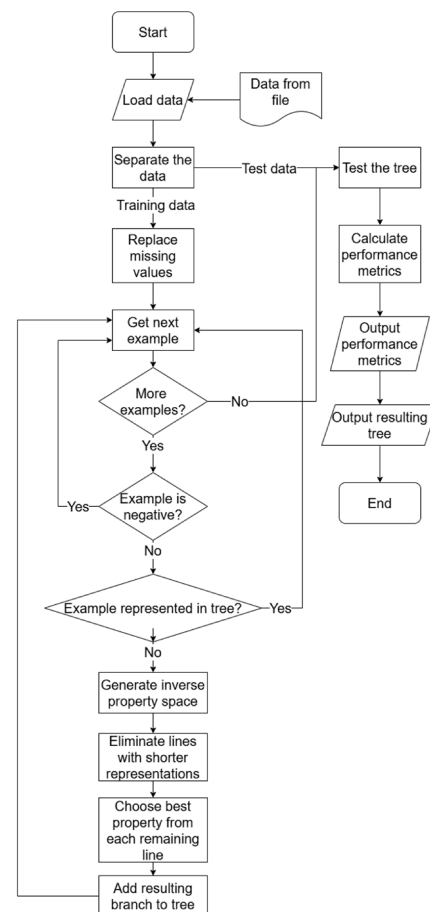


Figure 1. Flowchart of the implementation.

Missing values in the training set are handled by replacing them with the most frequent value observed at the corresponding attribute position among samples belonging to the same class as the incomplete instance.

4.2. Tree Building

The training examples are then processed sequentially. Since the objective is to construct a tree exclusively for positive examples, samples labeled as negative are disregarded. If a tree has already been constructed, it is first evaluated to determine whether it can classify the current example. If classification is not possible, an inverse property space is generated.

To construct the inverse property space, the class label of the considered example is inverted. The attribute values of this example are then compared with the corresponding values of all other examples. If the values match, they cannot serve as distinguishing features and are therefore discarded. If they differ, the value from the considered example is retained at that position.

Once the inverse property space has been generated, each entry is examined to determine whether a more concise equivalent representation exists. If a shorter representation is identified, the original entry is removed. This refinement process ensures that only minimal representations are preserved. As a result, the final inverse property space contains exclusively the shortest meaningful representations associated with the current example.

To determine the most suitable properties for constructing the branch that classifies the current example, each property remaining in the inverse property space is evaluated by counting how many positive examples in the original property space share the same

value. For each entry, the value with the highest frequency is selected and incorporated into the branch.

This heuristic approach eliminates the need to expand the inverse property space into a full logical expression, which could be computationally expensive for large expressions. Finally, the constructed branch is integrated into the tree structure, and the procedure is repeated for the next example.

5. Experiments and Results

This section presents the experimental results obtained from evaluating the proposed method on the selected datasets.

5.1. The MONK Problems

The MONK problems [18] are a set of standardized machine learning benchmarks designed to evaluate the performance of algorithms in solving binary classification tasks. The data is an artificial domain in which robots are described by six discrete attributes:

- a1: head_shape {round, square, octagon}
- a2: body_shape {round, square, octagon}
- a3: is_smiling {yes, no}
- a4: holding {sword, balloon, flag}
- a5: jacket_color {red, yellow, green, blue}
- a6: has_tie {yes, no}

There are 432 possible combinations of these attributes. The classes of the samples are determined by 3 sets of increasingly complex rules. The training subsets are pre-selected for these sets.

5.1.1. MONK-1

This dataset is intended to evaluate the algorithm's ability to capture non-linear patterns, where the target class is defined by the equality of two specified attributes. The data is noise-free. The classification rules are as follows: (head_shape = body_shape) or (jacket_color = red). The training set comprises 124 randomly selected examples. The results are presented in Figure 2.

Predicted

R e a l		+	-
	+	216	0
	-	0	216

Figure 2. MONK-1 confusion matrix.

Performance metrics are as follows:

- Accuracy Rate: 100%;
- Error Rate: 0%;
- True Positive Rate: 100%;
- True Negative Rate: 100%;
- Precision Rate: 100%.

For the first dataset, the method constructs a tree that correctly classifies all possible examples, achieving 100% accuracy.

5.1.2. MONK-2

This dataset involves a more complex logical rule. The data is noise-free. The classification rule specifies that exactly two of the attributes must take their first value. The training set consists of 169 randomly selected examples. The results are presented in Figure 3.

Predicted

R e a l		+	-
	+	86	56
	-	53	237

Figure 3. MONK-2 confusion matrix.

Performance metrics are as follows:

- Accuracy Rate: 74.7685%
- Error Rate: 25.2315%
- True Positive Rate: 60.5634%
- True Negative Rate: 81.7241%
- Precision Rate: 61.8705%

5.1.3. MONK-3

This dataset incorporates noise through the inclusion of mislabeled examples. The classification rules are defined as follows: (jacket_color = green and holding_sword = yes) or (jacket_color $\neq$ blue and body_shape $\neq$ octagon). The training set consists of 122 randomly selected examples, of which 5% are mislabeled. The results are presented in Figure 4.

Predicted

R e a l		+	-
	+	162	66
	-	40	164

Figure 4. MONK-3 confusion matrix.

Performance metrics are as follows:

- Accuracy Rate: 74.7685%;
- Error Rate: 24.537%;
- True Positive Rate: 71.0526%;
- True Negative Rate: 80.3922%;
- Precision Rate: 80.198%.

5.2. Congressional Voting Records

The Congressional Voting Records dataset [19] is a suitable benchmark for the task of classification. It contains voting records from the United States House of Representatives, documenting how members of Congress voted on a set of key legislative issues.

Each instance in the dataset represents an individual Congress member and their corresponding votes (“yes” or “no”) on 16 distinct bills. The dataset comprises 435 records in total, of which 267 are labeled as Democrat and 168 as Republican. Each record is described by 16 binary-valued attributes reflecting the member’s position on each legislative measure.

For the purposes of this study, two separate experiments are conducted. In the first experiment, the majority class (Democrat) is designated as the target class and a classification model is constructed accordingly. In the second experiment, the class labels are inverted, and the model is trained with Republican as the target class.

Additionally, the results reported herein are an average of ten consecutive runs of the implementation. This measure is intended to avoid any misleading results stemming from the random separation of the data into training and testing subsets.

5.2.1. Congressional Voting Records—Target Democrat

The average of the results is presented in Figure 5.

Predicted

R e a l		+	-
	+	82.6	3.2
	-	14.2	42.2

Figure 5. Congressional voting records—target Democrat confusion matrix.

Performance metrics are as follows:

- Accuracy Rate: 87.64%
- Error Rate: 12.36%
- True Positive Rate: 96.31%
- True Negative Rate: 73.15%
- Precision Rate: 86.60%

5.2.2. Congressional Voting Records—Target Republican

The average of the results is presented in Figure 6.

Predicted

R e a l		+	-
	+	51.8	2.7
	-	7.7	78

Figure 6. Congressional voting records—target Republican confusion matrix.

Performance metrics are as follows:

- Accuracy Rate: 92.68%
- Error Rate: 7.32%
- True Positive Rate: 95.08%
- True Negative Rate: 91.24%
- Precision Rate: 87.91%

The results show that the proposed method correctly classifies the examples with high accuracy and precision. Even in data that includes noise, for example, through incorrectly labeled examples, a working rule for recognizing the classes is extracted.

6. Discussion and Conclusions

This article presents a new heuristic algorithm for conceptual learning. The decision tree is constructed through the minimization of a logical function. More precisely, the heuristic approach eliminates the need to expand a logical expression, which could be computationally expensive for large expressions. Tests conducted on various sets of standardized machine learning benchmark datasets show that the algorithm operates with high accuracy and precision on noise-free data, as well as on datasets that include noise through mislabeled examples. The algorithm's ability to capture non-linear patterns was experimentally evaluated, where the target class is defined by the equality of two specified attributes.

The acquired knowledge, formally represented as logical rules, can be integrated into hybrid neuro-symbolic architectures with the aim of achieving improved machine learning performance, more reliable responses to user queries without hallucinations, learning with less data, and better-grounded and more interpretable machine learning and machine reasoning.

The proposed heuristic differs from classical top-down induction methods such as C4.5 [11] and CART [10] in its treatment of the property space. These algorithms rely on greedy recursive partitioning based on statistical criteria, which may lead to complex trees that are subsequently simplified through pruning. Rule extraction is typically performed after tree construction, resulting in representations whose size depends on the number and depth of leaves. In contrast, the proposed method integrates logical function minimization and value prevalence directly into the construction process, targeting compact covering rules without requiring full tree expansion.

From a computational perspective, the method has linear space complexity with respect to the number of features and samples, as no fully expanded intermediate structures are maintained. While the theoretical time complexity may become exponential in worst-case scenarios—particularly for continuous attributes or features with high cardinality—this behavior was not observed in the conducted experiments. Across the evaluated datasets, the algorithm exhibited stable runtimes comparable to classical decision tree methods, suggesting that the effective search space remains tractable in practice. This empirical result supports the claim that avoiding explicit logical expansion can yield computational savings without sacrificing interpretability, aligning with common preprocessing practices in algorithms such as C4.5 [11] and with broader objectives in Explainable Artificial Intelligence.

The proposed algorithm for conceptual learning can be applied to machine learning in an adaptive environment that combines artificial intelligence, machine learning, and virtual reality. The adaptive environment in virtual reality encompasses simulations and intelligent training platforms. These systems adapt in real time to the user's behavior, skills, and environment. Machine learning in such an adaptive environment is dynamic, adjusting continuously to new data and user behavior.

The proposed algorithm can support multiple possible adaptation policies that are compatible with the user's behavior. It can maintain multiple hypotheses, observe the user, and eliminate inconsistent hypotheses, thereby narrowing the hypothesis space.

It is also possible to maintain a set of policies, each with a probabilistic weight, instead of relying on a single fixed model. Rather than maintaining just one model, multiple models can be used—for example, a skill model, a cognitive load model, and an action-prediction model. Then, depending on the context, a different model can be selected.

In this way, the applications of these systems include virtual reality therapeutic systems, autonomous interactive environments, autonomous driving systems, and intelligent training simulators.

Author Contributions: Conceptualization, D.M. and D.B.; methodology, D.M., R.T. and D.B.; software, D.M.; formal analysis, D.M. and D.B.; writing—original draft preparation, D.M., R.T. and D.B.; writing—review and editing, D.M. and D.B.; supervision, R.T.; funding acquisition, R.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research is realized and funded under the scientific-research project № КП-06-ПН97/47 “Exploring the possibilities of using artificial intelligence and simulation models to increase energy efficiency” by the contract КП-06-Н97/11 with the Bulgarian National Science Fund.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in the study are openly available in UCI Machine Learning Repository at <https://doi.org/10.24432/C5R30R> and <https://doi.org/10.24432/C5C01P> or ref. [18,19].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

EBL	Explanation-based learning
EAI	Explainable Artificial Intelligence
LLM	Large Language Model
FOL	First-Order Logic
DNF	Disjunctive normal form

References

1. CLRN Team. What Is Concept Learning? California Learning Resource Network. Available online: <https://www.clrn.org/what-is-concept-learning/> (accessed on 6 March 2026).
2. von Rueden, L.; Mayer, S.; Beckh, K.; Georgiev, B.; Giesselbach, S.; Heese, R.; Kirsch, B.; Walczak, M.; Pfrommer, J.; Pick, A.; et al. Informed Machine Learning—A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 614–633. [CrossRef]
3. Besold, T.R.; Garcez, A.D.; Bader, S.; Bowman, H.; Domingos, P.; Hitzler, P.; Kühnberger, K.-U.; Lamb, L.C.; Lima, P.M.V.; de Penning, L.; et al. Neural-Symbolic Learning and Reasoning: A Survey and Interpretation. *arXiv* **2017**, arXiv:1711.03902. [CrossRef]
4. Dash, T.; Chitlangia, S.; Ahuja, A.; Srinivasan, A. A review of some techniques for inclusion of domain-knowledge into deep neural networks. *Sci. Rep.* **2022**, *12*, 1040. [CrossRef] [PubMed]
5. Attention Is All You Need | Proceedings of the 31st International Conference on Neural Information Processing Systems. Guide Proceedings. Available online: <https://dl.acm.org/doi/10.5555/3295222.3295349> (accessed on 6 March 2026).
6. Mustafa, F.; Dine, F. Deep Learning for Knowledge Representation: Automating Semantic Analysis and Ontology Construction. 2025. Available online: <https://doi.org/10.13140/RG.2.2.35145.45928> (accessed on 24 January 2025). [CrossRef]
7. Phiri, T. Reasoning Agents: Building Chatbots That Think Step-by-Step—ChatNexus. Available online: <https://articles.chatnexus.io/knowledge-base/reasoning-agents-building-chatbots-that-think-step/> (accessed on 6 March 2026).
8. Xu, C.; Lan, T.; Ji, Y.; Yu, C.; Wang, W.; Gao, J.; Dong, Q.; Qian, K.; Li, P.; Bi, W.; et al. DECIDER: A Dual-System Rule-Controllable Decoding Framework for Language Generation. *arXiv* **2025**, arXiv:2403.01954. [CrossRef]
9. Dakovski, L.G.; Shevked, Z. Alternative approach for learning from examples. In Proceedings of the CompSysTech’2005, Varna, Bulgaria, 16–17 June 2005; pp. IIIB.5-1–IIIB.5-6.
10. Breiman, L.; Jerome, F.; Olshen, R.A.; Charles, J.S. *Classification and Regression Trees*, 1st ed.; Chapman and Hall/CRC: New York, NY, USA, 2017. [CrossRef]
11. Quinlan, J.R. *C4.5: Programs for Machine Learning*; Morgan Kaufmann Publishers, Inc.: San Francisco, CA, USA, 1993.
12. Clark, P.; Niblett, T. The CN2 Induction Algorithm. *Mach. Learn.* **1989**, *3*, 261–283. [CrossRef]
13. Cohen, W.W. Fast Effective Rule Induction. In *Proceedings of the Twelfth International Conference on International Conference on Machine Learning (ICML’95)*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1995; pp. 115–123.
14. Ruggieri, S. Complete Search for Feature Selection in Decision Trees. *J. Mach. Learn. Res.* **2019**, *20*, 1–34.

15. Alvarez, J.M.; Scott, K.M.; Berendt, B.; Ruggieri, S. Domain Adaptive Decision Trees: Implications for Accuracy and Fairness. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*; Association for Computing Machinery: New York, NY, USA, 2023; pp. 423–433. [[CrossRef](#)]
16. Craven, M.; Shavlik, J. Extracting Tree-Structured Representations of Trained Networks. In *Proceedings of the Advances in Neural Information Processing Systems 8, NIPS, Denver, CO, USA, 27–30 November 1995*. Available online: https://proceedings.neurips.cc/paper_files/paper/1995/file/45f31d16b1058d586fc3be7207b58053-Paper.pdf (accessed on 6 March 2026).
17. Wang, Z.; Zhang, W.; Liu, N.; Wang, J. Scalable Rule-Based Representation Learning for Interpretable Classification. *NeurIPS* **2021**, 30479–30491. Available online: https://proceedings.neurips.cc/paper_files/paper/2021/file/ffbd6cbb019a1413183c8d08f2929307-Paper.pdf (accessed on 11 March 2026).
18. Wnek, J. MONK's Problems [Dataset]. (1993). UCI Machine Learning Repository. Available online: <https://doi.org/10.24432/C5R30R> (accessed on 24 January 2025).
19. Congressional Voting Records [Dataset]. (1987). UCI Machine Learning Repository. Available online: <https://doi.org/10.24432/C5C01P> (accessed on 24 January 2025).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.