Journal Title
XX(X):1–33
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Neurosymbolic Inference, Reasoning and Proving in Propositional Logic: A Survey

Nenad Krdzavac [1] and Felix Engel [1]

Abstract

Neurosymbolic AI frameworks aim to combine neural networks with symbolic inference, reasoning, and proving, yet their reasoning and proving capabilities and assurance in propositional logic (PL) remains limited. In this survey, we introduce the X–Y–Z taxonomy, implemented as a instance of SKOS ontology, designed to evaluate NeSy frameworks by their levels of reasoning and proving capability and assurance. To ensure reproducibilty of the evaluation NeSy frameworks, we implemented a Python evaluation pipeline that computes aggregate statistics such, as mean, median, and entropy, over surveyed NeSy frameworks. The Python project is publicly available for extension or updates. By offering both a classification tool and an executable evaluation methodology, this work facilitates direct comparison across NeSy frameworks, guides the development of systems that balance capability with assurance, and advances the maturity of the NeSy discipline.

Keywords

Neurosymboic AI, Propositional Logic, Inference, Reasoning, Proving

1 Introduction

Neurosymbolic (NeSy) AI has achieved impressive capabilities by combining the learning strategies with the formal inference, reasoning and proving methods. Despite these advances, current NeSy frameworks exhibit a critical gap: while neural components excel at learning from data, they often lack guarantees for formal reasoning, and proving capabilities, as well as assurance levels.

To address this gap, this survey proposes a systematic methodology for evaluating NeSy frameworks with respect to their reasoning and proving capabilities and the assurance they provide. We introduce the X–Y–Z taxonomy, implemented as an instance of SKOS ontology. Using this taxonomy, we implemented a fully automated, reproducible evaluation pipeline in Python that computes aggregate statistics over surveyed frameworks, allowing researchers to measure and compare capabilities and assurance levels consistently.

This survey is organized as follows. After the Introduction, Section 2 presents the propositional logic (PL), as well inference, reasoning, and proving tasks in PL. It also introduces the neurosymbolic (NeSy) reasoning paradigm and its approaches, along with a brief overview of artificial neural networks. Section 3 describes X–Y–Z taxonomy used to survey NeSy frameworks. Related work on existing survey papers in NeSy AI is reviewed in Section 4. Section 5 surveys literature on NeSy AI that employs inference, reasoning, and proving in PL. An evaluation of NeSy frameworks is presented in Section 6, and Section 7 concludes the paper.

2 Background

2.1 Propositional Logic

In this section, propositional logic (PL) is introduced before addressing its inference, reasoning, and proving tasks. The PL is based on propositions. Not all sentences are qualified as propositions. For example, questions or commands are excluded. The *alphabet* of PL consists of four pairwise disjoint sets

$$\Sigma = P \cup M \cup O \cup C. \tag{1}$$

The set P is defined as a countable infinite set of *atomic propositional letters*, also called *atomic words* [52], *simple or atomic propositions* [4, p. 36, Definition 2.2], *propositional variables* [145, p. 30, Section 2,1], written as p_1, p_2, p_3, p_4, $\ldots$, p_n, etc. The set M of *punctuation marks* contains the parentheses "(" and ")". The set O includes the logical connectives symbols, including conjunction ($\wedge$), disjunction ($\vee$), negation ($\neg$), implication ($\rightarrow$), and equivalence ($\leftrightarrow$)

[1]TIBLeibniz Information Centre for Science and Technology, Hanover, Germany

Corresponding author:
Nenad Krdzavac, TIBLeibniz Information Centre for Science and Technology, Hanover, Germany.
Email: Nenad.Krdzavac@tib.eu

[52]. The set C of logical constants comprises the symbols $\top$, and $\bot$.

Each propositional variable, such as p_1, is a *well-formed formula* (w.f.f.) in PL [4]. If p_1 is a w.f.f. then its negation $\neg p_1$ is also a w.f.f. A *literal* is defined as a propositional variable or its negation [145, p. 99, Section 3.3.1]. Similarly, if p_1 and p_2 are w.f.f., then the formulas $(p_1 \vee p_2)$, $(p_1 \wedge p_2)$, $(p_1 \rightarrow p_2)$, and $(p_1 \leftrightarrow p_2)$ are w.f.f. as well [4]. We use only aforementioned rules to construct the w.f.f. in PL. As an illustration, let p_1 denote the sentence *"it is raining"* and let p_2 denote the sentence *"the umbrella is open"*. Then the formula $(p_1 \rightarrow p_2)$ is a w.f.f., expressing *"if it is raining, then the umbrella is open"*. As another example, the PL formula $(p_1 \wedge \top)$ is a w.f.f., whereas the PL expression $(p_1 \vee \top \vee)$ is not, since it does not follow rules for constructing w.f.f in PL [4].

In PL, a truth assignment is defined by an *interpretation*, which is a function $I : P \rightarrow \{\top, \bot\}$ [94, Section 5.1.2] that maps a set of propositional variable to $\top$ or $\bot$ [4]. Formally, this interpretation (also called valuation) is precise and rigorously specified [4, p. 37, Definition 2.1.1]. Let p_1 denote the sentence *"it is raining"*. The truth value of this proposition may vary with context and therefore be subjective [52]. The interpretation function I assigns truth values to any w.f.f. in PL [37, p. 20, Section 1.2]. Truth tables provide a systematic method for determining the truth value of PL statements. Let p_1 and p_2 be atomic propositions. Then the interpretation function assigns *true* to $\neg p_1$ if and only if (iff) it assigns *false* to p_1. For $p_1 \wedge p_2$, the interpretation function assigns *true* only when both p_1 and p_2 are *true*, otherwise the formula is false. Similarly, for $p_1 \vee p_2$, the interpretation function assigns *true* iff at least one of them is *true*, and *false* otherwise. The proposition $p_1 \rightarrow p_2$ is assigned *false* only when p_1 is *true* and p_2 is *false*. In all other cases, it is true. Finally, proposition $p_1 \leftrightarrow p_2$ is assigned *true* iff p_1 and p_2 have the same truth value, and *false* otherwise. In addition to truth value semantics, PL also admits an algebraic semantics [8], which is not covered in this work.

Any w.f.f. in PL can be rewritten in either conjunctive normal form (CNF) or disjunctive normal form (DNF). A CNF formula is a conjunction of disjunctions of literals (see Formula 2), whereas a DNF formula is a disjunction of conjunctions of literals (see Formula 3) [74, Section 3].

$$(p_1 \vee p_2) \wedge (\neg p_3 \vee p_4) \tag{2}$$

$$(p_1 \wedge \neg p_3) \vee (p_1 \wedge p_4) \vee (p_2 \wedge \neg p_3) \vee (p_2 \wedge p_4) \tag{3}$$

PL clause is a disjunction of literals. Horn clause in PL is a spacial form of CNF containing at most one positive literal [74, p. 267, Section 3]. For example, formula (4) is a Horn clause.

$$(\neg p_1 \vee p_2 \vee \neg p_3) \tag{4}$$

Horn formula is a conjunction of Horn clauses [74, p. 268, Section 3].

In the following three sections, 2.2, 2.3, and 2.4 inference, reasoning, and proving in PL are described. Understanding these three concepts is crucial for understanding inference, reasoning, and proving in NeSy frameworks. Figure 1 shows a Venn diagram which illustrates inference, reasoning, and proving approaches in PL. Section 2.5 describes AI reasoning paradigms referenced in the remaining part of this survey.

2.2 Inference in Propositional Logic

In PL, an inference denotes a formal mechanism for deriving a PL statement, called the conclusion, from one or more PL statements, called the premises. Such a mechanism is specified by an inference rule, which defines a form of derivation [37, p. 110]. An inference rule is said to be truth-preserving or valid if, whenever all premises are true, the conclusion must also be true. An example of an inference rule is *Modus ponendo ponens* (MPP) [105], which is formally expressed as

$$\frac{P, \quad P \rightarrow Q}{Q} \tag{5}$$

where P, Q are atomic propositions. It states that if the atomic proposition P is true ($\top$), and the implication $P \rightarrow Q$ is also true, then the atomic proposition Q must be true. This property can be verified using truth tables. In contrast, the argument form (6) is not a valid inference rule in PL because it is not truth-preserving. In particular, there exists a truth assignment to P for which the premise is true while the conclusion is false.

$$\frac{P}{\bot} \tag{6}$$

It is important to note that the validity of an inference rule depends on the rule itself, not on the truth values assigned to the atomic propositions. Even if one or more of the premises is false, denoted by $\bot$, the conclusion may be either true or false. However, the inference rule itself remains truth-preserving. Resolution step (hereafter simply *resolution*) in PL is another fundamental inference rule in PL [145, p. 98–99, Section 3.3.1]. It can be expressed as follows:

$$\frac{(P \vee Q), (\neg P \vee R)}{(Q \vee R)} \tag{7}$$

where P, Q, and R are atomic propositions. In resolution, it is allowed to use negation of atomic propositions, such as $\neg P$. The resolution rule is truth-preserving, meaning that no truth assignment to atomic propositions can be found such that the premises are true and the conclusion is false.

The next section discusses reasoning in PL and clarifies the distinction between inference and reasoning within PL. It also highlights how, in the NeSy AI AI literature, the terms inference and reasoning are often used interchangeably.

2.3 Reasoning in Propositional Logic

The preceding section delineates inference in PL, namely, the utilization of inference rules such as MPP and resolution. It is a rule-based mechanism that derives a conclusion from given premises. By contrast, reasoning in PL refers to an algorithmic process that applies fixed, truth-preserving inference rules. This process relies on additional mechanisms such as the ordering of rule applications, termination criteria, search strategies, and redundancy elimination.

Truth tables in PL are not reasoning methods in themselves. Rather, they provide a (semantic) procedure for evaluating the truth values of propositional formulas under all possible interpretations. By examining a truth table, one can determine fundamental logical properties such as satisfiability (SAT), tautology (validity), and logical entailment.

Logical entailment in PL, written as $p_1 \models p_2$, is a semantic property of PL. It means that in every interpretation in which p_1 is true, p_2 is also true.

A PL expression (8) represents one w.f.f. Using truth tables as reasoning method, it is possible to determine which assignments make this w.f.f. *true*.

$$(p_1 \vee p_2) \wedge (\neg p_1 \vee \top) \tag{8}$$

The truth assignment of the w.f.f. (8) depends on the truth values assigned to the propositional variables p_1 and p_2. As shown in Table 1, the interpretation functions assigns *true* to w.f.f. (8) when p_1 is *false* and p_2 is *true*. This demonstrates that w.f.f. (8) is *satisfiable* [52], since there exists an truth value assignment to the propositional variable that makes the formula *true*.

Table 1. Truth-value assignments of $(p_1 \vee p_2) \wedge (\neg p_1 \vee \top)$.

p_1	p_2	$\neg p_1$	$p_1 \vee p_2$	$(p_1 \vee p_2) \wedge \neg p_1$
T	T	F	T	F
T	F	F	T	F
F	T	T	T	T
F	F	T	F	F

There are several variants of SAT problems, including Max-SAT [77, Section 2], 3SAT or 2SAT [65, Section 2.1]. Max-SAT problem involves finding a truth value assignment to propositional variables that maximizes the number of satisfied clauses in a CNF formula. In other words, a clause is considered satisfied if at least one of its literal is true [77, Section 2]. In 3SAT problem, we are given set of propositional variables and a collection of clauses, where each clause contains exactly three literals. The 3SAT task is to determine whether there exists a truth assignment for the finite set of propositional variables that satisfies all clauses in the collection [47, p. 46, Section 3.1]. A similar definition applies to 2SAT problem in PL [65, Section 2.1].

Resolution method as syntactic reasoning technique in PL [55, p. 179, Section 3.11]. By contrast, the tableau method is a semantic reasoning technique in PL that provides a decision procedure for the SAT problem [63, p. 29, Section 2.3, Definition 2.10]. It incrementally constructs a tree structure for a PL formula by systematically decomposing it into simpler components using tableau rules [63, p. 29, Section 2.3, Definition 2.13].

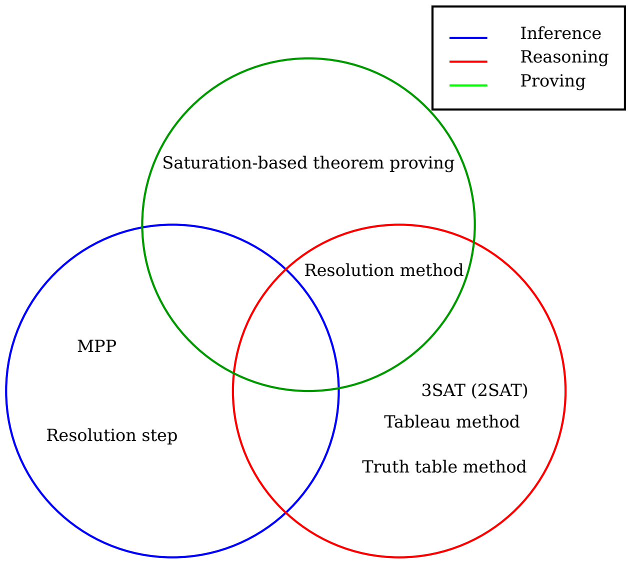

Figure 1. Venn diagram illustrating inference, reasoning, and proving approaches in PL.

In PL, a w.f.f. is a tautology if it is true under every possible truth assignment to its atomic propositions [37, p. 23]. Consequently, a w.f.f. is a contradiction if it is false under every possible truth assignment to its atomic propositions.

2.4 Theorem Proving in Propositional Logic

Theorem proving in PL concerns determining whether a PL formula is valid, that is, whether its negation is unsatisfiable (UNSAT) [93, see Propositional Proof Procedure Section]. Two representative approaches to theorem proving in PL are saturation-based method [10, see Introduction section], and and resolution-based theorem proving [93, see Resolution in Propositional Calculus Section]. Notably, the resolution method, as illustrated in the Venn diagram in Figure 1, can be viewed both as a reasoning method and as a theorem proving technique. However, the formulation of the resolution method presented in [93, Resolution in Propositional Calculus Section] is specifically developed for theorem-proving purposes.

2.5 NeSy AI Reasoning

In this section, we describe AI reasoning paradigms, with a particular emphasis on the NeSy AI reasoning as presented in the survey paper [73]. These paradigms are referenced throughout the remainder of this paper. Figure 2 illustrates the categorization of AI reasoning proposed in the survey [73, Section 2.2.2.].

NeSy AI reasoning is characterized by the integration of logic-based rules within a neural network. As described in the survey paper [73, Section 2.2.1, formula 6], NeSy reasoning can be formalized as a function that takes two inputs: an observation, presented as a query or a graph encoded by a neural encoder, and knowledge, expressed as rules, ontologies, or graphs. In the case of PL, rules can be expressed either as PL implications or in CNF. The reasoning output is then derived jointly from both the neural and symbolic components. In PL case, rules can be expressed in the form of PL implications or in CNF.

Deductive, inductive, and abductive reasoning are classified under the task structure category in Liang's taxonomy (see Figure 2). Deductive reasoning derives logically valid conclusions from known premises or rules [73, Section 2.2.2]. In contrast, inductive reasoning generalizes from specific instances to broader rules or models [73, Section 2.2.2]. In both cases, premises, rules, or instances can be expressed in PL, typically using implications. Abductive reasoning, on the other hand, seeks the most plausible explanation for a given observation [73, Section 2.2.2]. Its formalization in PL is provided in [95, p. 705, Definition 3.2].

Neural reasoning is classified under the representation type of reasoning in Liang's taxonomy (see Figure 2). It relies on learned representations within a neural network. Unlike formal logic reasoning, neural reasoning does not use explicit rule structures. The inference outcome is predicted on multi-layer transformations and pattern abstraction [73, Section 2.2.2]. Essentially, it performs pattern-based inference rather than reasoning in the traditional formal logic sense. In some approaches, neural reasoning can also leverage graph-based embeddings of PL formulas [49].

Hybrid reasoning in NeSy AI falls under representation type category in Liang's taxonomy (see Figure 2) [73]. It leverages the flexibility to unify the interpretability of formal logic with the learning capabilities of neural networks. As a result, hybrid reasoning performed by NeSy frameworks can be understood and explained in human-readable terms.

In the literature on NeSy AI, the terms *inference* and *reasoning* are used interchangeably. For example, in [38], the authors use the term inference to describe a specific algorithm, named as *forward and backward chaining inference*, which implements logical reasoning over Horn clauses expressed in first order logic. In the standard AI literature, such as in [103], forward chaining is explicitly described as reasoning algorithm for knowledge bases written in Horn form [103, p. 275]. These two naming conventions do not conflict. Reasoning refers to the abstract logical process of deriving conclusions [103], whereas inference denotes the concrete algorithmic procedure that executes this reasoning within a NeSy system [38].

2.6 A Short Introduction to Neural Networks

Artificial neural networks (ANNs) [53] serve as the neural component in a NeSy framework and can be viewed as simplified models of the networks of neurons in the human brain. A simple artificial neuron is illustrated in [53, Figure 1.2]. In ANNs, synapses are represented by single numerical weights that multiply each input before it reaches the neurons equivalent of the cell body. The weighted inputs are then summed using arithmetic addition. One of the simplest models of an artificial neuron is the Threshold Logic Unit (TLU) [53, Section 1.1], which computes the weighted sum of its inputs and generates an output based on a threshold. A network of such artificial neurons forms an artificial neural network [53].

3 A Taxonomy for NeSy Frameworks Survey

This section describes a taxonomy for categorizing and evaluating NeSy frameworks that perform reasoning and proving tasks in PL. Known as the X–Y–Z taxonomy, it is structured along three dimensions:

1. *X:* the level of reasoning and proving capability in PL.
2. *Y:* the level of reasoning and proving assurance in PL.
3. *Z:* the evaluation metrics used to assess NeSy frameworks.

Together, these dimensions provide a structured basis for the systematic comparison of NeSy frameworks and for identifying limitations in existing approaches. By imposing a clear organizational structure, the taxonomy supports research toward more reliable and verifiable NeSy systems, particularly for complex reasoning and proving tasks.

The taxonomy is implemented as an instance of a SKOS ontology [82]. It employs SKOS

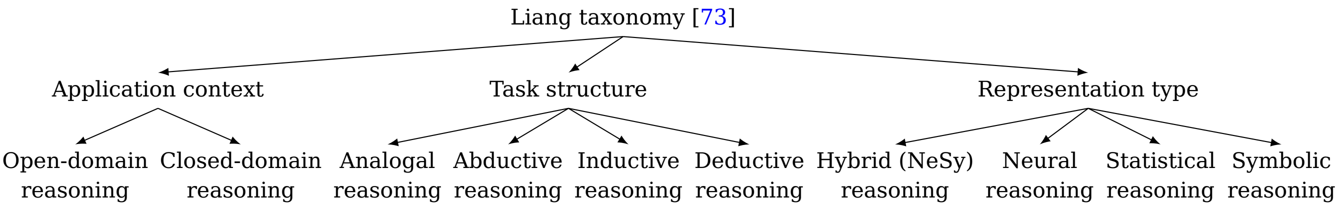

Figure 2. Tree-shaped taxonomy for categorizing AI reasoning with focus on NeSy AI [73]

broader and *narrower* relations to organize concepts hierarchically. The implementation of taxonomy as a TTL file is available at the URL provided in Listing 3 in Appendix A.1.

3.1 X: NeSy Reasoning and Proving Capability Level in PL

Figure 3 illustrates the X dimension of the taxonomy, which characterizes the level of reasoning and proving capability in PL supported by a NeSy framework. This dimension captures how effectively a system can perform reasoning or proving tasks in PL. The X dimension is organized into eight groups: foundational reasoning (X1.1), witness construction (X1.2), strategic reasoning and adaptation (X1.3), search and control (X1.4), training and robustness (X1.5), integration and expressivity (X1.6), verification support (X1.7), and robustness and interpretability (X1.8).

```
X1:Reasoning and proving capability level
    X1.1: Foundational reasoning
        X1.1.1: Decision mechanisms
            X1.1.1.1: Basic decisions
                X1.1.1.1.1: Decision-only (SAT/UNSAT)
        X1.1.2: Logical inference (derivation of relations)
    X1.2: Witness construction
        X1.2.1: Constructive witness generation
    X1.3: Strategic reasoning and adaptation
        X1.3.1: Logical strategies
            X1.3.1.1: Reasoning methods
                X1.3.1.1.1: Core propositional SAT solving
                X1.3.1.1.2: Resolution
            X1.3.1.2: Heuristic adaptation
                X1.3.1.2.1: Meta-reasoning and strategy adaptation
                X1.3.1.2.2: Reinforcement learning for clause selection
                X1.3.1.2.3: Query-aware formula generation
        X1.3.2: Optimization extensions
            X1.3.2.1: MaxSAT optimization
    X1.4: Search and control
        X1.4.1: Lemma discovery and reuse
        X1.4.2: Clause selection quality
        X1.4.3: Pruning and control of search space
        X1.4.4: Branching heuristics
        X1.4.5: Heuristic optimization for SAT
    X1.5: Training and robustness
        X1.5.1: General logical reasoning
        X1.5.2: Paradox-resistant training
        X1.5.3: Core UNSAT prediction and refutation
    X1.6: Integration and expressivity
        X1.6.1: Logical integration
         X1.6.1.1: Logic mechanisms
            X1.6.1.1.1: Algorithmic alignment
            X1.6.1.1.2: Logic expressivity & proof calculus support
    X1.7: Verification support
        X1.7.1: Certificate evaluation
            X1.7.1.1: Soundness & verified guarantees
    X1.8: Robustness and Interpretability
        X1.8.1: Robustness
        X1.8.2: Interpretability
            X1.8.2.1: Explainability & human interpretability
```

Figure 3. X dimension: taxonomy for NeSy reasoning and proving capability level

Foundational reasoning comprises *decision mechanisms* (X1.1.1) and *logical inference* (X1.1.2) concepts. Decision mechanisms include the *decision-only (SAT/UNSAT)* capability (X1.1.1.1.1), which refers to a NeSy framework's ability to efficiently return a binary SAT/UNSAT result for a PL formula without providing proof, explanation, or certificate. In contrast, logical inference concept denotes the application of valid inference rules to derive conclusions from one or more premises.

At the same time, witness construction includes the *constructive witness generation* mechanism (X1.2.1), a core capability of NeSy frameworks for PL reasoning. This mechanism generates verifiable truth assignment that serves as evidence of a formula's is satisfiability.

Strategic reasoning and adaptation (X1.3) encompasses advanced NeSy techniques aimed improving the efficiency and scalability of reasoning in PL. This dimension includes two main categories *logical strategies* (X1.3.1) and *optimization extensions* (X1.3.2). Logical strategies comprise reasoning methods such as

1. *Core propositional SAT solving* (X1.3.1.1.1)
2. *Resolution* (X1.3.1.1.2)

In addition, *heuristic adaptation* methods (X1.3.1.2), provides more flexible reasoning approaches, including

1. Meta-reasoning and strategy adaptation (X1.3.1.2.1)
2. Reinforcement learning for PL clause selection (X1.3.1.2.2)
3. Query-aware formula generation (X1.3.1.2.3)

Optimization extensions focus on enhancing reasoning strategies and include *MaxSAT optimization* (X1.3.2.1).

Core propositional SAT solving (X1.3.1.1.1) refers to a NeSy frameworks ability to solve standard SAT problems in PL. It serves as a baseline for evaluating the frameworks reasoning and proving capability level. In the other hand, *resolution* (X1.3.1.1.2) is a reasoning method for determining SAT by deriving contradictions from PL formula. *Meta-reasoning and strategy adaptation* (X1.3.1.2.1) refers to a set of capabilities that allow a NeSy framework to dynamically enhance its efficiency in PL reasoning and proving tasks. For example, the framework can adapt its learning strategies for branching heuristics when solving the SAT problem of a PL formula. *Reinforcement learning (RL) for clause selection*

(X1.3.1.2.2) concept frames clause selection as a key task for RL in NeSy theorem proving. In this settings, RL can identify novel and effective strategy for selecting clause [118]. *Query-aware formula generation* (X1.3.1.2.3) leverages the query mechanism within a NeSy framework to solve SAT problems in PL [87].

Optimization extensions (X1.3.2), including *MaxSAT optimization* (X1.3.2.1), involve solving the MaxSAT problem [146] within NeSy reasoning, allowing the framework to find assignment that satisfy the largest number of clauses.

Search and control (X1.4) encompasses five key capabilities that strengthen NeSy reasoning and proving capability level: *lemma discovery and reuse* (X1.4.1), *clause selection quality* (X1.4.2), *pruning and control of search space* (X1.4.3), *branching heuristics* (X1.4.4), and *heuristic optimization for SAT* (X1.4.5) techniques. *Lemma discovery and reuse* (X1.4.1) explains how a NeSy framework can generate new lemmas during the proof process. Specifically, lemma reuse refers to the frameworks ability to store these lemmas and apply them later during proof search and reasoning. *Clause selection quality* (X1.4.2) in the context of NeSy automated theorem proving in PL, refers to a frameworks ability to choose clauses during proof search in a way that minimizes the search space and reduces proof time [117]. The concept of *pruning and control of search space* (X1.4.3) concept addresses the SAT problem by eliminating unsatisfiable branches from the search space [58]. In NeSy frameworks, it refers to mechanisms that guide or restrict the exploration of the combinatorial search space during reasoning or proof in propositional logic (PL). *Branching heuristics* (X1.4.4) refers to an emerging heuristic approach used by NeSy frameworks to learn how to solve the propositional SAT problem [110]. *Heuristic optimization for SAT* (X1.4.5) refers to methods used by NeSy frameworks to solve SAT problem. These methods can be classified into global and local heuristic optimization approaches [108].

Training and robustness (X1.5) encompasses key concepts such as *general logical reasoning* (X1.5.1), *paradox-resistant training* (X1.5.2), and *core UNSAT prediction and refutation* (X1.5.3). Specifically, *general logical reasoning* (X1.5.1) refers to NeSy frameworks capable of performing deductive reasoning on domain agnostic tasks [27]. Building on this, *paradox-resistant training* (X1.5.2) ensures that the reasoning mechanisms avoid generating paradoxes during training. Finally, *core UNSAT prediction and refutation* (X1.5.3) focuses on frameworks where the neural part component predicts which PL clauses belong to an UNSAT core, followed by explicit refutation or validation by a PL solver [109].

Integration and expressivity (X1.6) covers *logic mechanisms* (X1.6.1.1), including *algorithmic alignment* (X1.6.1.1.1) and *logic expressivity & proof calculus support* (X1.6.1.1.2). Algorithmic alignment concept refers to the degree to which learning approach is compatibile with PL syntax. In contrast, logic expressivity & proof calculus support concept describes the range of proof methods the framework can represent and manipulate [137].

Verification support (X1.7) refers to mechanisms within frameworks that ensure the correctness of results by providing evidence that can be independently checked to confirm system correctness. Within the X–Y–Z hierarchy, *certificate evaluation* (X1.7.1) is a more specific concept under verification support. Within certificate evaluation, *soundness & verified guarantees* (X1.7.1.1) concept provides formal assurances of correctness ensuring that the results are trustworthy.

Robustness (X1.8.1) refers to the ability of a NeSy framework to maintain stable performance [12]. *Interpretability* (X1.8.2) denotes the extent to which a human can understand and reason about internal structure and decisiom-making process of the NeSy framework [89]. This concept has more specific concept named *explainability & human interpretability* (X1.8.2.1), where explainability focuses on generating human understandable output of NeSy framework.

3.2 *Y: NeSy Reasoning and Proving Assurance Level in PL*

This section describes Y dimension of X–Y–Z taxonomy named as reasoning and proving assurance level in PL. In the taxonomy, *verification and certification* (Y1.1) concept ensures the correctness of proof through mechanisms such as *certificate support*(Y1.1.1), *proof validation* (Y1.1.2), and *refutation certification* (Y1.1.3). These processes enable checkable, machine verifiable soundness for NeSy reasoning and proving assurance level.

```
Y1: Reasoning and proving assurance level
    Y1.1: Verification and certification
        Y1.1.1: Certificate support
            Y1.1.1.1: Certificate format support (DRAT/DRUP)
            Y1.1.1.2: Provable soundness via external verifier
        Y1.1.2: Proof validation
            Y1.1.2.1: Proof compression
        Y1.1.3: Refutation certification
            Y1.1.3.1: Refutation or UNSAT certification
    Y1.2: Explainability for assurance
        Y1.2.1: Interactive explainability
    Y1.3: Core soundness guarantee
        Y1.3.1: Soundness
```

Figure 4. Y dimension: taxonomy for NeSy reasoning and theorem proving assurance level

Certificate support (Y1.1.1), within the verification and certification group, is defined when a NeSy framework and system can produce machine-checkable and human-interpretable proofs, as well as compress proofs when applicable. Certificate support also encompasses interactive explainability, which enables humans or external tools to inspect and validate reasoning steps, thereby enhancing transparency and trust between NeSy AI and humans. The certificate support has two more specific concepts such as *certificate format support (DRAT/DRUP)* (Y1.1.1.1) and *provable soundness via external verifier* (Y1.1.1.2).

Certificate format support (Y1.1.1.1) requires the NeSy system to output proofs in the Deletion Resolution Asymmetric Tautology (DRAT) format, a generalization of the earlier Deletion Reverse Unit Propagation (DRUP) format [139].

Provable soundness via external verifier (Y1.1.1.2), however, ensures that these proofs can be independently verified for soundness by any external verifier, such as DRAT-trim [69].

Proof validation (Y1.1.2) in NeSy reasoning and proving assurance refers to the process in which the neural component of a NeSy framework generates a reasoning trace, while the symbolic component automatically validates its logical correctness. Within the X–Y–Z taxonomy, the *proof validation* concept is broader than the *proof compression* (Y1.1.2.1) concept. Proof validation may also involve tasks such as redundancy elimination within proofs.

Proof compression (Y1.1.2.1) refers to a method for efficient handling of formal proof in PL [102] [51]. Various compression techniques have been developed to compress the size of an already existing proof in PL, thereby improving the efficiency of proof verification. Foundational studies, such as [102], have specifically examined resolution-based proof compression in PL.

Refutation or UNSAT certification (Y1.1.3.1) refers to a NeSy framework capability to generate a verifiable and complete UNSAT proof certificate (e.g., in DRAT format (Y1.1.1)) during the proving process. These certificates can then be verified by an external verifier (Y1.1.2) to confirm the UNSAT result [96]..

Interactive explainability (Y1.2.1) is a narrower concept of *Explainability for assurance* (Y1.2) in the X–Y–Z taxonomy, which generates step-by-step, human-interpretable reasoning traces. Unlike explainability approaches that focus on a PL formula of minimal size that produces the same truth value as the input PL formula [62], interactive explainability emphasizes explanations in natural language combined with symbolic traces [9]. This concept is crucial for the NeSy AI because it enhances the interpretability of reasoning processes for users and fosters trustworthy collaboration between humans and AI.

Soundness (Y1.3.1) of a NeSy proof system refers to the property that if the system can prove a PL formula from a set of PL formulas, then that formula is indeed entailed by the set [115]. If a NeSy framework is sound than the framework cannot derive false statements from true premises.

3.3 Z: Taxonomy for Evaluating NeSy Frameworks

This section describes Z dimension of X–Y–Z taxonomy we use to evaluate NeSy frameworks. In the taxonomy concepts Z1.1 and Z1.2 represent numerical values for reasoning and proving capability (X dimension) and assurance (Y dimension) levels. In contrast, concept Z1.3 represents the integration level of neural learning with inference, reasoning, or proving methods in PL.

```
Z1: Evaluation metrics
    Z1.1: Capability level
    Z1.2: Assurance level
    Z1.3: Integration level
```

Figure 5. Z dimension: taxonomy for evaluating NeSy frameworks

4 Existing Surveys

This section provides an overview of existing survey and review papers on NeSy AI. In particular, it examines the taxonomies proposed in these works and discusses how NeSy frameworks are classified using these taxonomies with respect to reasoning and proving capabilities and assurance levels in PL. The analysis is based on publications retrieved via a SPARQL query executed against the Digital Bibliography & Library Project (DBLP) SPARQL endpoint. The query and its results are publicly available in a Gitlab repository, referenced in Listing 2 in the Appendix A.1.

In total, 76 papers published between 1971 and 2025 were retrieved and analyzed. Of those, 25 are survey or review papers focused on NeSy frameworks. Table 2 summarizes the classification taxonomies used in 25 survey and review papers on NeSy AI and their alignment with the X–Y–Z taxonomy.

Figure 6 shows a Venn diagram that illustrates survey papers that use more than one classification system to categorize NeSy frameworks, based on data in Table 2. In addition, the X and Y dimensions are also used to highlight the NeSy reasoning and proving capability or assurance level in PL for the research papers cited by the surveyed and reviewed literature. Notably, some survey and review articles listed in Table 2, such as [29], do not clearly explain the reasoning and proving capability or assurance level in PL that are already discussed in the primary works they reference. The alignment between classifications (taxonomies) employed in these survey papers and the X and Y dimensions address this gap.

Table 2. Taxonomies employed in survey papers on NeSy AI. Alignment taxonomies used in the survey papers with X–Y–Z taxonomy.

ID	Taxonomy name	Survey papers	Dimensions within taxonomy	X and Y concept
1.	PRISMA methodology [88] [81]	[29]	1. Logic and reasoning 2. Learning and inference 3. Explainability & trustworthiness 4. Knowledge representation	X1.1.1 X1.3.1.2.1 Y1.2.1 Y1.3.1
2.	Task-directed NeSy taxonomy	[35]	1. Rule mining 2. Rule enforcement 3. Program synthesis	Y1.2
3.	H. Kautz's taxonomy [64]	[34] [41] [133] [97] [54] [17] [1] [68]	1. K–1:Symbolic Neuro Symbolic 2. K–2:Symbolic[Neuro] 3. K–3:Neuro|Symbolic 4. K–4:Neuro:Symbolic $\rightarrow$ Neuro 5. K–5:Neuro_[Symbolic] 6. K–6:Neuro[Symbolic]	See Table 4
4.	ESI nomenclature	[54, p. 25, Figure 28]	1. Ensemble 2. Sequential 3. Integrated 3.1 Nested 3.2 Coopeartive 3.3 Compiled	×
5.	LLR taxonomy	[34]	1. Logically informed embedding approaches 2. Learning With Logical Constraints 3. Rule learning for knowledge graph completion	See Table 4
6.	KR taxonomy	[28]	1. Knowledge graph completion 2. Rule learning dimension	X1.1, X1.1.1.1.1, X1.3.2.1, X1.3.1.1
7.	Natural language processing taxonomy	[3]	1. Tasks 2. Paradigm	X1.1.1, X1.1.1.1, X1.5.1, X1.3.1.1
8.	SWOT methodology	[85]	1. Strengths 2. Weaknesses 3. Opportunities 4. Threats	X1.1.1, X1.1.1.1.1 X1.2.1, X1.3.1 Y1.3.1
9.	G-L based representation	[19]	1. Graph-based representation 2. Logic-based representation	×
10.	A3PO taxonomy	[72]	1. Autoformalization 2. Premise selection 3. Proofstep generation 4. Proof search 5. Other tasks	×
11.	NeSy four-dimensional taxonomy	[136]	1. Functionality 2. Knowledge embedding 3. Knowledge representation 4. Neural-symbolic integration	×
12.	NeSy-7 taxonomy	[78]	1. The approach to logical inference. 2. The syntax of the used logical theories. 3. The logical semantics of the systems and their extensions to facilitate learning. 4. The scope of learning, encompassing either parameter or structure learning. 5. The presence of symbolic and subsymbolic representations. 6. The degree to which systems capture the original logic, probabilistic, and neural paradigms. 7. The classes of learning tasks the systems.	×
13.	Yu's taxonomy	[142] [97] [54] [1]	1. Learning for reasoning 2. Reasoning for learning 3. Learning-reasoning	See Table 5 See Table 6
14.	KLREA taxonomy	[50]	1. Knowledge representation 2. Learning 3. Reasoning 4. Explainability and trustworthiness 5. Applications	×

Table 3. Taxonomies employed in survey papers on NeSy AI. Alignment taxonomies used in the survey papers with X–Y–Z taxonomy. (continue)

ID	Taxonomy name	Survey papers	Dimensions within taxonomy	X and Y concept
15.	×	[16]	×	×
16.	×	[15]	×	×
17.	KRERL taxonomy	[44, p. 627, Figure 5]	1. Knowledge Representation 2. Extraction 3. Reasoning 4. Learning	×
18.	ADT taxonomy	[123]	1. Quality 2. Expressive power 3. Translucency of the network 4. Portability 5. Complexity	×
19.	ILU taxonomy	[11] [1] [21]	1. Interrelation 2. Language 3. Usage	×

The Preferred Reporting Items for Systematic Reviews and Meta-Analyzes (PRISMA) 2020 framework is a widely adopted methodology for reporting a systematic review of a study [88]. According to the most recent iteration of the PRISMA 2020 statement, the checklist comprises 27 items, while the abstract section is subject to 12 additional requirements [81]. The objective of these modifications is to ensure the transparency of scholarly review methodologies. Using the PRISMA methodology, paper [29] conducted a systematic review of NeSy AI projects published between 2020 and 2024. The review is structured along four fundamental dimensions, as outlined in Table 2. The authors placed particular emphasis on logic and reasoning within the broader context of NeSy AI. In the context of the aforementioned dimension, the authors undertake a comprehensive examination of eight NeSy frameworks [29, p. 6, Figure 2]. The following trends in NeSy AI research were identified as being of particular significance for logic and reasoning:

1. The integration of formal logic and probabilistic models is essential for supporting more robust decision-making processes (X1.1.1: *Decision mechanisms*; Y1.3.1: *Soundness*).
2. The incorporation of metacognitive mechanisms should facilitate the adaptation of NeSy frameworks in their reasoning strategies (X1.3.1.2.1:*Meta-reasoning and strategy adaptation*). This adaptation should enhance the interpretability and clarity of explanations (Y1.2.1: *Interactive explainability*).

However, the review does not offer a detailed assessment of which form of PL reasoning or proving methods (X1.3.1.1:*Reasoning methods*) nor which type of formal verification (Y1.1:*Verification and certification*) are actually supported by the reviewed NeSy frameworks. For example, the authors cite [24] to illustrate the integration of NeSy frameworks with autonomous systems, but in the same work [24], to support decision-making, authors implemented a resolution procedure with non-binary unification (X1.3.1.1.2:*Resolution*) and use a NeSy theorem prover to query formulas expressed in a formal logic and to conduct reasoning (X1.5.1:*General logical reasoning*; X1.3.1.1:*Reasoning methods*) over these formulas derived from text written in a natural language.

The task-directed NeSy taxonomy proposed in [35] classifies NeSy approaches into three categories: rule mining, rule enforcement, and program synthesis. This taxonomy provides the organizing structure for surveying NeSy frameworks published between 2017 and 2024. A central research question in the survey concerns how symbolic components can be integrated with neural networks such as solvers to enhance explainability (Y1.2:*Explainability for assurance*) and reasoning capabilities. The survey does not focus exclusively on NeSy reasoning or theorem proving in PL. In the context of rule mining, the survey highlights the role of Horn clauses, originally developed in PL, which serve to represent structured knowledge and enable reasoning across domains such as medicine. The survey also discusses *propositionalization*, understood as the transformation of relational data into vectorized representations suitable for neural learning. Crucially, *propositionalization* is not equivalent to translating relational data into PL formulas. Within the rule enforcement category, the survey discusses NeSy regularization strategies [2], as well as reasoning approaches based on first-order logic [13, p. 17, Section 3] and probabilistic logic [86], particularly in the context of enhancing explainability. Finally, within the program synthesis category, the survey describes examples of using the Prover9 solver [80] in semantic parsing pipelines.

According to Kautz's taxonomy, NeSy architectures are classified into six types [64]. Instead of providing a comprehensive account of the six types of NeSy architectures, for a more thorough examination of this topic, please refer to the concise summary in the first paragraph of Section 3.3 in [19]. For a more detailed exposition of this topic, please refer to the original work [64]. As illustrated in Table 2, eight scholarly articles are enumerated that adapt Kautz's taxonomy (from K–1 to K–6) to survey NeSy frameworks. An exhaustive exposition of the mappings between Kautz's taxonomy and the X and Y concepts is provided in Table 4, encompassing these seven survey papers.

A comprehensive survey into the application of NeSy AI methodologies for reasoning over knowledge graphs is presented in the study [34]. The authors propose a novel taxonomy to describe NeSy AI approaches for reasoning over knowledge graphs. As illustrated in Figure 1 of the survey [34] and listed in row 4 of the Table 2, this taxonomy is organized into the following three main categories:

A. Logically informed embedding approaches.
B. Learning with logical constraints.
C. Rule learning for knowledge graph completion.

Each category is comprised of one or more subgroups, which correspond to specific types of Kautz's taxonomy. To reference these subgroups in this paper, the notation *category_id(K–n)* is used, where n denotes the associated type in Kautz's taxonomy. For instance, *A(K–3)* denotes a group of NeSy frameworks within Kautz's type K–3 category that corresponds to category *A*. In the background section, the survey describes PL Horn clauses. Within the A(K–4) subgroup, authors highlight how Horn clauses and the forward-chaining method (X1.1:*Foundational reasoning*, X1.3.1.1:*Reasoning methods*) are used in knowledge graph augmentation [34, Subsection IV-A.2]. The authors discuss the application of Horn clauses in rule mining [34, Subsection IV-C]. There is no evidence in the survey paper [34] that Horn clauses expressed in PL play a role in rule mining or knowledge graph augmentation. Within the B(K–6) subgroup, authors cited work on the graph collaborative reasoning method [26] which translates knowledge graph structure into formulas in a formal logic [34, Subsection IV-B.2]. In the survey [34] authors do not mention that these formulas translated from knowledge graphs are expressed in PL. Theorem 1 in [26, p. 77, Theorem 1] guarantees when these PL formulas are true or false (X1.3.1.1: *Reasoning methods*). However, the survey [34] does not examine how theorem proving methods can be applied to knowledge graphs in a NeSy theorem proving context. In the broader NeSy sense, the application of theorem proving techniques in knowledge graphs still remains largely underexplored, particularly in the context of verifying the soundness (Y1.3.1:*Logical strategies*). The utilization of such methodologies has the potential to facilitate tasks such as rule guided embeddings [34, Subsection IV-A.2] by facilitating the implementation of Horn constraints solving methods [131]. Furthermore, the integration of proof compression techniques (Y1.1.2.1:*Proof compression*) can directly reduce the size of derived proofs in PL. As demonstrated in [57], causal-proof trimming and related optimizations yield reduced proof in PL [57, Section III]. These optimizations have the potential to enhance the interpretability and verifiability of reasoning processes applied to knowledge graphs. Also, an important work on theorem proving over knowledge graphs [71], introduces a KG-Prover, which integrates a mathematical knowledge graph with a large language model and the Lean proof assistant [33] to generate and verify formal proofs (Y1.1.2:*Proof validation*).

In the paper [41], the author undertakes an extensive review of NeSy AI techniques and their applications in signal and image processing. The review is organized across Kautz's six dimensions (see Table 2, and Table 4), the basis of which is the integration level between symbolic reasoning and neural networks. In the review of NeSy AI applications in multimedia, vision, and entertainment, the author describes a NeSy framework, called Laser [61]. This framework aligns raw video clips with PL for the purpose of improving semantic video understanding. However, in this specific instance, the author does not expound upon the manner in which NeSy reasoning in PL is manifested within the Laser framework. The present review makes no mention of NeSy theorem proving in PL.

The paper [133] provides a concise survey of cognitive AI systems, focusing on NeSy AI algorithms. While Kautzs taxonomy [64] defines six NeSy architectures, the survey employs a subset of five categories to structure its analysis of NeSy AI algorithms. This classification serves as a useful entry point for readers seeking to navigate the landscape of NeSy algorithms. Although the survey does not undertake a comprehensive examination of NeSy reasoning capabilities. Additionally, it does not address NeSy theorem proving as a discrete research domain. For instance, within the K–5 category, the authors cite [44], but they do not discuss the propositionalization component emphasized in that paper. The importance of explainability (X1.8.2.1: *Explainability & human interpretability*) and robustness (X1.8.1: *Robustness*) of symbolic reasoning is discussed within the K–6 category. Despite the significant pertinence of the work to the realm of reasoning-centric NeSy research, the treatment of these mechanisms in the survey is cursory, thereby failing to furnish

Table 4. Mapping between categories of Kautz's taxonomy and the X and Y concepts across survey papers that employ Kautz's taxonomy.

ID	Reference	K–1	K–2	K–3	K–4	K–5	K–6	Comment
1	[34]	×	×	×	X1.1, X1.3.1.1	×	X1.3.1.1	
2	[41]	×	×	×	×	×	×	
3	[133]	×	×	×	×	×	X1.8.1, X1.8.2.1,	
4	[97]	×	×	X1.3.1.1, X1.1.1.1.1 Y1.3.1 X1.7.1.1	×	×	X1.1.1, X1.1.1.1, X1.3.1.1	
5	[54]	×	×	×	×	×	×	
6	[17]	X1.1	×	×	X1.1.1.1	×	×	see Table 8
7	[68]	×	×	×	×	×	×	

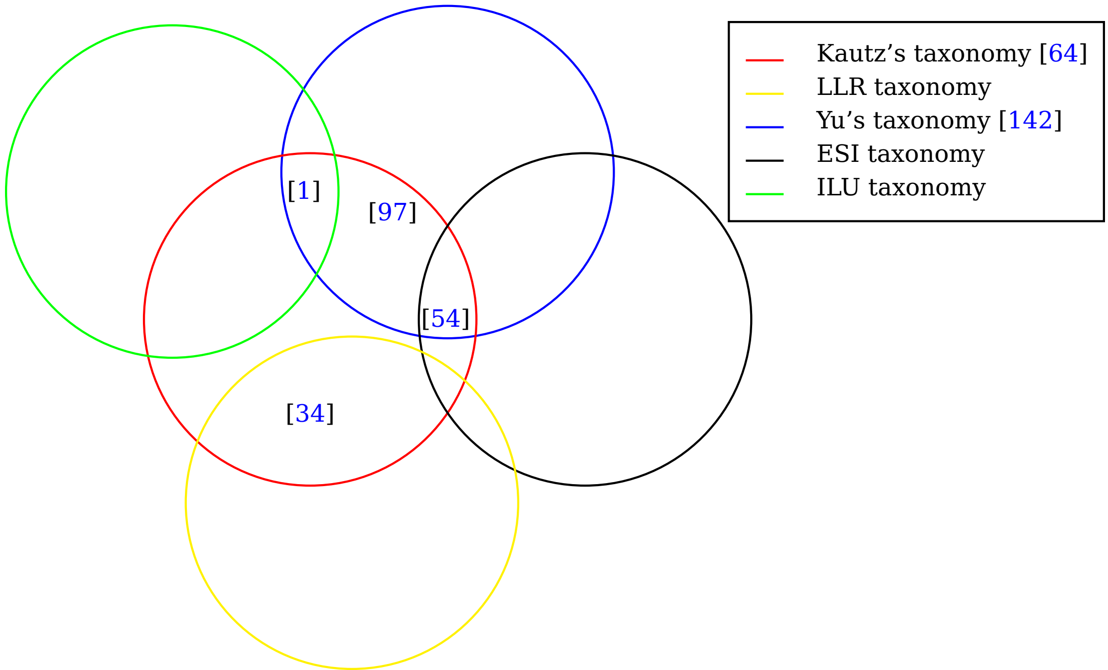

Figure 6. Venn diagram illustrating survey papers that use multiple taxonomies to categorize NeSy frameworks, derived from Table 2).

readers with a comprehensive grasp of the specific mechanisms underpinning NeSy reasoning.

The paper [68] adopts Kautz's taxonomy to review the use of graph neural networks (GNNs) as a model for NeSy computing, including the review of the application of GNNs in different domains. However, inference, reasoning, and proving in PL are not explicitly discussed within categories of the Kautz's taxonomy (see Table 4), but they are discussed in the context of the GNNs relationship to NeSy computing [68, p. 2-5, Section 3].

However, Table 2 lists four papers that utilize Yus taxonomy [142]. Table 5, in turn, illustrates methods that utilize PL in NeSy frameworks described in [142]. Table 6 presents a detailed mapping between Yus taxonomy and the X and Y concepts across survey papers that adopt Yus taxonomy. The Yus taxonomy is predicated on the integration mode between neural and symbolic systems. The tree in Figure 7 presents Yus taxonomy, which classifies NeSy learning frameworks into three distinct categories: learning for reasoning, reasoning for learning, and learningreasoning [142, p. 109, Table 2]. Each category is associated with one or more methods, represented as leaf nodes in the tree shown in Figure 7.

Table 5. Methods that utilize PL in NeSy frameworks within categories across Yu's taxonomy [142].

ID	Reference	Learning for Reasoning	Reasoning for Learning	Learning-Reasoning
1	[140]	×	reguarization (semantic loss)	×
2	[113]	×	transfering	×

In survey [142, p. 113-114, Section 3.2.1.], the authors describe semantic loss as a regularization approach within reasoning for learning category (see

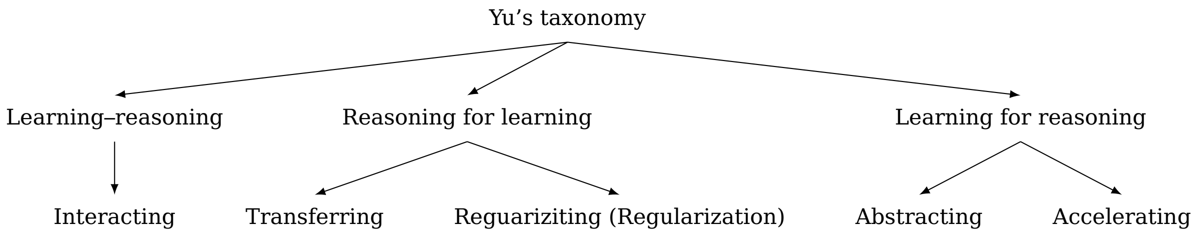

Figure 7. Methods assigned to categories in Yu's taxonomy, derived from [142, p. 109, Table 2]

Figure 7), stating that it utilizes PL reasoning (X1.1: *Foundational reasoning*, see Table 6) to improve the learning ability of neural networks [140]. However, the survey does not mention which PL reasoning methods are employed. A closer look at the original paper that describes semantic loss reveals that the PL statement, expressed in CNF, is used as input to a weighted model counting (WMC) [14, p. 4, Section 2.2] to compute semantic loss [140, p. 114, Section 5.1, Figure 3]. From a formal logic perspective, WMC is a PL inference method rather than a reasoning method [25].

The survey [1] uses Yu's taxonomy to categorize NeSy reinforcement learning frameworks. However, as illustrated in Figure 2, the survey also highlights Kautz's taxonomy [1, p. 1942, see Table I, Table II]. The survey describes work on Propositional Logic Nets (PROLONETS) [113] that encodes domain knowledge written in PL into a neural network, but the authors do not describe the reasoning and proving capability or assurance level of such NeSy reinforcement learning frameworks, which is marked as × in Table 6. However, in survey [142, p. 116-117, Section 3.2.1., Figure 13] authors describe NeSy framework [113] that utilizes a transferring method within the reasoning for learning category of Yu's taxonomy (see Table 5), and the authors claim that PROLONETS encodes PL statements in neural networks to improve neural network reasoning capability level.

Table 6. Mapping between Yu's taxonomy and the X and Y concepts across survey papers that employ Yu's taxonomy [142].

ID	Reference	Learning for Reasoning	Reasoning for Learning	Learning--Reasoning
1	[142]	×	X.1.1	×
2	[1]	×	×	×
3	[97]	X1.1.1, X1.1.1.1, X1.3.1.1	×	X1.1.1, X1.1.1.1

The survey [97] aligns the taxonomies proposed by Kautz's [64] and Yu's [142] to describe how NeSy frameworks support *validation & verification*, as well as *testing & evaluation*. Table 7 provides a comprehensive overview of the aforementioned alignment, presenting the NeSy frameworks belonging to categories of Kautz's and Yu's taxonomies, as categorized in the [97] survey. For instance, the original work [76] introduces DeepProbLog as a programming language that integrates probabilistic logic inference and deep learning techniques.

Table 7. NeSy frameworks classified according to both Kautz's [64] and Yu's [142] taxonomies, as reported in survey [97].

ID	Kautz's taxonomy	Yu's taxonomy	NeSy framework
1.	K-2:Symbolic[Neuro]	Learning for Reasoning	AlphaGo Zero [114]
2.	K-3:Neuro|Symbolic	Learning for Reasoning or Learning-Reasoning	DeepProbLog [76]
3.	K-6:Neuro[Symbolic]	Learning for Reasoning or Learning-Reasoning	Shield [5]

However, [142, p. 5, Table 2] survey classifies DeepProbLog under the *Learning-Reasoning* category, whereas [97, p. 10] places it in the *Learning for Reasoning* category. This flexibility in categorizing NeSy frameworks underscores the interpretative nature of such assignments, which often supersedes absolute interpretation. Readers, including PhD students and practitioners, should treat these classifications as informed estimations or approximations, reflecting the reasoning behind why a particular NeSy framework fits into a certain category. Since different authors may apply different criteria, mismatches in taxonomy assignments are to be expected.

The survey [97, p. 2] characterizes *validation & verification* as processes that ensure software meets required quality and reliability standards (Y1:*Reasoning and proving assurance level*). The *testing & evaluation* are described as methods for conducting these processes (X1.7:*Verification support*). Within this context, PL is introduced as a logical foundation that supports *validation & verification* (X1.1:*Foundational reasoning*; X1.6.1.1:*Logical Integration*). To establish this connection with precision, the authors map *validation & verification* tasks to corresponding reasoning mechanisms in PL (X1.6.1.1: *Logic mechanisms*). The validation of the symbolic component of NeSy AI is tied to logical concepts in PL such as validity, soundness (Y1.3.1:*Soundness*; X1.7.1.1:*Soundness & verified* *guarantees*). Other PL related concepts, including decidability, truth tables, and semantic tableau, are also discussed in [97, Sections IV-A]. Notably, semantic tableau in PL can be mapped both to fundamental decision procedures in PL (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*) and to reasoning methods (X1.3.1.1: *Reasoning methods*) concepts. These concepts are also used in the *verification* of symbolic part of NeSy AI [97, Sections IV-B]. In summary, the survey is pertinent to the broader AI community because it illuminates the manner in which reasoning mechanisms in PL support validation and verification of NeSy frameworks. As an illustrative example, authors cite DeepProbLog [97, p. 10] to describe that semantic tableau in PL (X1.3.1.1: *Reasoning methods*) can be applied within the verification process. While the survey clearly demonstrates the role of PL-based reasoning in NeSy frameworks such as DeepProbLog, this connection becomes less explicit when considering other NeSy approaches discussed in the survey. Following the categorization reported in the survey [97], the work [5] is classified under the K–6 and Learning for Reasoning (Learning-Reasoning) categories (see Table 7). Based on classification of NeSy frameworks in [97], mappings from corresponding K–6 and Learning for Reasoning (Learning-Reasoning) categories to the X and Y concepts are illustrated in Table 4 and Table 6. In the aforementioned work [5], a *reactive system*, known as *Shield*, is utilized as a computational model that monitors the learning process and performs corrective actions when a chosen action violates a predetermined learning specification [5, see Abstract]. Notably, the survey does not address the fact that propositions in PL are used to model the inputs and the outputs of the *reactive system* [5, p. 3]. The truth value assignment (X1.1.1:*Decision mechanisms*; X1.1.1.1:*Basic decisions*) is employed to validate the inputs and the outputs of the reactive system.

The structured review [54] examines NeSy frameworks for natural language processing (NLP). The review aims to answer the question of whether these NeSy frameworks meet the authors' descriptions and promises, which include reasoning, out-of-distribution generalization, interpretability, effective learning and reasoning from limited data, and transferability across domains [54, see Abstract]. The review incorporates Kautz's taxonomy within a nomenclature (see Table 2) to categorize NeSy frameworks. However, the review highlights Yu's taxonomy. Readers are referred to [54, Figures 28–34, p. 2528] that provides clear and unambiguous visualizations of all categories within the nomenclature used for categorizing NeSy frameworks for the NLP. In the review, authors clearly divide NeSy reasoning into model-based and theorem proving approaches [54, p. 8]. However, authors reveal in the review that there are discrepancies in the definition of reasoning and variation in its implementation within NeSy AI. Authors cite early NeSy frameworks such as KBANN [120] and CILP [32] that embed PL in neural network [54, Figures 28–34, p. 7]. However, authors cited SHRUTI [138] and Neural Theorem Prover (NTP) [100] as NeSy theorem proving frameworks, but authors did not describe what type of proving these frameworks can perform.

The survey [17] provides an overview of NeSy implementations, categorizing them by integration type, application domain, system properties, and contribution category, as illustrated in [17, p. 12813, Figure 3]. Reasoning paradigms such as deductive, inductive, combinatorial, common sense, and abductive [45] are described under the system properties category [17, p. 12817, Figure 6]. To organize NeSy implementations by integration type, the authors adopt Kautz's taxonomy [64]. For each category in this taxonomy, the survey describes how individual NeSy frameworks employ these reasoning methods. Based on categorization given in [17], Table 8 summarizes the reasoning paradigms, marked as $\times$ if reasoning paradigm is not given, the number of cited works that are associated with PL across Kautz's taxonomy. Moreover, the table illustrates the mappings from PL-based reasoning methods utilized by these NeSy frameworks to X and Y concepts. Overall, Table 8 shows that the survey [17] cites 28 papers that are associated with PL, either as a representational language or as an active component of the NeSy reasoning process.

The authors highlight that K–1 type NeSy frameworks employ deductive [17, p. 12819, Section 2.3.1] and common sense [17, p. 12826, Table 7] reasoning methods (see ID 1 in Table 8). However, the role of PL in NeSy reasoning is not explicitly discussed within the K–1 category of Kautz's taxonomy that is quantified with zero in 4th column of Table 8. The survey also does not cite any NeSy frameworks that leverage a formal logic for deductive reasoning or in connection to common sense reasoning [17, p. 12826, row 4 in Table 7]. Notably, the survey cites [116] as a K–1 type of NeSy framework. However, it does not mention the use of PL reasoning or representation in that framework. In contrast, [116] clearly describes using PL as basis for reasoning in this framework by translating PL into a neural network implementation (X1.1: *Foundational Reasoning*) [116, p. 2]. Therefore, the fifth column in Table 8 contains the reference to [116] to reflect its use of PL. The authors cite eight NeSy frameworks within the K-2 category and claim that they use PL [17, p. 12827, Table 8]. Among these eight citations, two cited NeSyframeworks thats do not have associated reasoning methods (marked as $\times$ in Table 8 and in [17, p. 12827, Table 8]). As illustrated in Table 8, three NeSy frameworks are

Table 8. Reasoning paradigms or methods utilized in NeSy frameworks across Kautzs taxonomy based on survey [17] .

ID	Category in Kautz's taxonomy	Inference / reasoning paradigm / method	Number of NeSy frameworks cited in [17] that are associated with PL	NeSy frameworks cited in [17] that use PL reasoning	X and Y concept
1.	K–1	Deductive	0	[116]	X1.1
		Common sense	0	×	×
2.	K–2	Case-based	6	×	×
		×	2	×	×
3.	K–3	Deductive	2	×	×
		×	1	×	×
4.	K–4	Constraint-based	0	×	×
		Case-based	2	×	×
		Neurule	1	×	×
		×	9	[56]	X1.1.1.1
5.	K–5	Case-based	2	×	×
		Deductive	1	×	×
		Common Sense	1	×	×
		×	1	×	×
6.	K–6	×	0	×	×
		TOTAL:	28		

cited in the survey [17, p. 12828, Table 9] and are thus classified within the K–3 category of Kautzs taxonomy. Of the three NeSy frameworks under consideration, two have been assigned to PL and employ a deductive reasoning paradigm. The survey cites [66] and asserts that the framework employs PL, however, the underlying reasoning paradigm remains unspecified [17, Reasoning column, row 1 makred as × in Table 9]. Notably, the NeSy frameworks associated with PL in the survey [17, p. 12827, Table 8; p. 12828, Table 9] do not actually employ PL reasoning methods. These examples demonstrate that readers should not equate the presence of symbolic components with a reasoning method in a NeSy framework. In the context of citing the NeSy frameworks in the aforementioned survey paper, they may be founded upon PL, even in the absence of actual support reasoning in PL. The K–4 category encompasses a total of 12 cited papers in the domain of NeSy AI, all of which are associated with PL [17, p. 12829, Table 10]. As shown in Table 8, nine of the twelve cited NeSy frameworks do not have an associated reasoning method. However, within the K–4 category, the authors cite the work in [56] and classify it as a NeSy framework without reasoning support. This assignment is questionable, as that work employs disjunction of the conjunctions (DNF) in PL to express rules and truth tables (X1.1.1.1: *Basic Decisions*), which function as a reasoning mechanism in the training set specification process [56, p. 7, Section 3.2]. In survey [17, p. 12829, Table 10], *Neurule* is categorized as an inference method (see Table 8). However, Neurule is a framework that employs a hybrid inference mechanism based on a backward chaining strategy [56, p. 11, Section 4.1, Section 4.3]. The survey cites five NeSy frameworks belonging to the K–5 category that are associated with PL [17, p. 12830, Table 11]. As illustrated in Table 8, four of these aforementioned NeSy frameworks have been assigned a reasoning paradigm or method. The remaining framework has not been assigned a reasoning paradigm and is indicated by an ×. Notably, none of the NeSy frameworks of type K–6 employ PL or reasoning methods in PL. In the context of Kautz's taxonomy, the existing literature includes studies on the encoding of PL within the neural architecture K_{BANN} [120] and on embedding PL utilizing the translation technique CIL^2P [46]. However, these two NeSy frameworks are not included in the Kautz's taxonomy [17, p. 12820, Table 6]. The survey paper makes no mention of theorem proving as a component of any NeSy framework.

Nevertheless, a thorough examination of NeSy knowledge graph reasoning methods is provided in [28], thereby complementing the previously discussed survey [34] (see ID 1 in Table 4). The authors of the study propose a taxonomy that is organized into two broad categories: knowledge graph completion and rule learning [28, p. 225:4, Figure 2]. The work systematically summarizes the intersection of neural networks and symbolic reasoning for knowledge graph reasoning tasks [28, p. 225:3, Table 1]. The survey also introduces PL (Boolean) reasoning (X1.1.1.1.1:*Decision-only*) as part of the symbolic reasoning section. It is acknowledged that PL possesses the theoretical capacity to support knowledge graph completion (X1.1:*Foundational reasoning*). The authors further highlight the role of the propositional maximum satisfiability problem (MaxSAT) [20, see Abstract] (X1.3.2.1:*MaxSAT optimization*) method as an optimization-oriented reasoning technique (X1.3.2:*optimization extensions*). The survey's findings within the rule learning category indicate that there is no evidence of the application of NeSy theorem proving techniques in the jointly learning embeddings of symbols and interpretable logical rules from knowledge graphs [28, p. 32].

A review of NeSy approaches in the domain of natural language processing (NLP) is presented by [3]. The authors have structured their review into *tasks* and *paradigm* dimensions, as illustrated in Figure 1 of their work [3]. Within the *paradigm* dimension, the review highlights methodologies that integrate a formal logic into neural networks to enhance NeSy reasoning capabilities (X1.5.1:*General logical reasoning*). For example, some approaches incorporate PL implication $A \rightarrow B$ with the goal of enabling neural networks to approximate truth value assignment (X1.1.1:*Decision mechanisms*; X1.1.1.1:*Basic decisions*). In another line of work, discussed by the authors, a proposition $A \wedge B$ is represented as a vector, and a neural network is then trained to learn the logical conjunction operation as a function, denoted as $f(A, B)$. This function effectively implements PL reasoning within a vector space (X1.3.1.1:*Reasoning methods*). By combining the reasoning power of symbolic logic with the flexibility and learning capacity of neural networks, these methods enable NeSy PL reasoning (X1.5.1:*General logical reasoning*) in NLP tasks.

As was the case in the preceding review, the present one addresses the impact of NeSy frameworks on advanced cognitive system in the domains of the NLP, including robotics, and decision-making [85]. The review employs the SWOT (strengths, weaknesses, opportunities, threats) methodology to analyze five extant NeSy AI models. In the context of NeSy reasoning and theorem proving, the present review focuses on NeSy reasoning based on first-order and probabilistic logics, as well as neural theorem proving. Nonetheless, the authors referenced work by [76] that employs DeepProbLog model. However, they do not delve into the role of PL reasoning within DeepProbLog. In the aforementioned work, the ground logic program is rewritten in PL to define the truth value (X1.1.1:*Decision mechanisms*) of the query in terms of the truth values of probabilistic facts [76, p. 7, Section 4.1]. In the review, the authors also emphasize the innovative nature of neural theorem proving methods, which integrate neural networks with theorem proving, contingent upon formal logic. However, the review does not emphasize the central of PL reasoning in neural theorem proving approaches (X1.1.1.1.1:*Decision-only*). The authors cited the work of [98] as being directly relevant to neural theorem proving. The research demonstrates how NeSy reasoning grounded in PL is embedded into the model architecture of logical neural networks (X1.2.1:*Constructive witness generation*; X1.3.1:*Logical strategies*). This integration enables the convergence of an inference algorithm with recursive directional graph traversal (Algorithm 3 in [98]) in finite time for the PL case (Y1.3.1:*Soundness guarantee*).

A survey of NeSy approaches for Network Intrusion Detection Systems (NIDS), accompanied by a critical analysis, is presented in [19]. The authors primarily employ a two-dimensional categorization scheme to differentiate NeSy frameworks, distinguishing between logic based and graph based representations. In the survey, authors describe the usage of PL in the NIDS frameworks to express rules in the *if-then* form that is PL implication ($p \rightarrow q$). For instance, authors cited a NeSy framework named ODXU [104] that utilizes PL to express decision trees, thereby enhancing the decision-making process. Nevertheless, the utilization of NeSy reasoning and proving capability or assurance level in PL within NIDS remains unaddressed.

The paper [72] presents a survey of deep learning for theorem proving. Authors highlight that Boolean satisfiability (SAT) solvers play a key role in theorem proving by handling PL, but in the rest of the paper authors did not added proving assurance level in PL supported by NeSy frameworks.

A four-dimensional taxonomy is employed in the survey by [136] to classify NeSy frameworks. One of these dimensions, knowledge representation, includes the use of PL. Notably, the survey does not specify how NeSy frameworks realize reasoning capabilities or achieve particular proving assurance levels. Instead, PL is discussed primarily in terms of its contribution to improving the performance of semi-supervised classification and its capacity to express rules.

The paper [78] surveys how reasoning and neural networks (called NeSy frameworks) are combined, as well as, how formal logic is combined with graphical probabilistic models (called StatAI frameworks). Authors use seven shared dimensions (see ID 12 in Table 2) to characterize these frameworks. The authors mentioned the SAT problem for PL [78, p. 4, Section 2] as a most fundamental one in computer science, but they did not explore how the reasoning and proving capability or assurance level in PL affects the properties of NeSy or StatAI frameworks.

The paper [50] surveys NeSy frameworks from five different angles (see ID 14 in Table 2). The authors explains that Logical Boltzmann Machines (LBM) [128] can represent any PL formula [50, p. 5, Section 2]. However, this topic is discussed in detail within the Section 5 of this paper. Overall, the authors did not discuss how well reasoning and proving capability or assurance level in PL is achieved in surveyed NeSy frameworks.

A comprehensive survey on NeSy learning and reasoning is given in [16]. PL is discussed for expressing problem in neural networks as a set of weighted PL formula [16, p. 23, Section 5.3.1].

However, NeSy reasoning and proving capability or assurance level in PL are not discussed.

Paper [15] survey NeSy approaches to visual query answering. The role of PL is seen as formal language to describe domain and encoding PL formula as a vector in graph convolutional network [15, p. 9, Section 4]. NeSy reasoning capability and proving level or assurance level in PL is not discussed.

The paper [44] reviews recent advances in NeSy computing, which is an approach to integrating machine learning with reasoning. The authors use a four-dimensional taxonomy to summarize NeSy techniques [44, p. 627, Table 5]. The *knowledge representation* dimension (see ID 17 in Table 3) includes PL as a formal language integrated within four NeSy frameworks. For example, authors describe how PL formulas are represented in neural networks such as [120], CILP [32]. This topic has already been discussed in the survey [54, Figures 28–34, p. 7].

The survey paper [123] does not directly employ PL. Instead, it reviews NeSy AI approaches for extracting relational explanations from deep neural networks. In the survey, the authors organize and compare knowledge-extraction methods according to five classification dimensions (see Table 3). The paper does not address the direct use of PL, nor does it discuss reasoning and proving capabilities and assurance levels. Logic programs are introduced only as a representation language, where clauses can be viewed, in restricted cases, as propositional implications.

A structured survey paper [11] reviews neural-symbolic (NeSy) frameworks along three classification dimensions (see Table 3). Along the *Language* dimension, PL is discussed as a knowledge representation language that can be embedded into neural networks. KBANN [121] s cited as a concrete example of encoding PL formulae within a neural architecture. Under *Usage* dimension, the authors discuss reasoning in PL within NeSy frameworks, but theorem proving capability and assurance level in PL is not addressed.

The paper [21] provides a comprehensive review of NeSy applications. In their survey, the authors adopt the three dimensional taxonomy introduced in [11]. PL is mentioned as one of the NeSy languages used in the domains covered. Nonetheless, the paper does not discuss the reasoning and theorem proving capabilities or the assurance levels in PL.

5 Literature Survey

This section surveys 43 research papers published between 1943 and 2025 placed chronologically in seven groups, as illustrated in Figure 9. Selected papers on NeSy frameworks combine PL and neural networks to perform inference, reasoning or proving tasks. The last column in the Table 9 shows the number of papers in each group. The assignment of X and Y concepts to individual NeSy frameworks is provided in tables within each subsection along with type of neural network architecture employed in NeSy framework.

5.1 1943–1989: Foundational Neural Logic Era

The association of PL with neural activation can be attributed to [79], whose work provided a foundational basis for later NeSy frameworks. Motivated by the *all-or-none* character of neural activity, authors proved that activation of neurons can be *treated using PL formula*. Instead of learning, their strategy relies on *rule-based* manual encoding: networks of neurons were converted into PL formulas (X1.6.1.1.2:*Logic expressivity & proof calculus support*; see Table 10). In the recursive construction [79, p. 8, Theorem III], each neuron N is assigned a unique numeral i, and synaptic connections between neurons are represented using logical connectives (X1.6.1.1.2:*Logic expressivity & proof calculus support*). To simulate reasoning, truth tables are used to evaluate all possible input combinations, assigning each proposition a truth value (T or F) (X1.1:*Foundational reasoning*). This work does not integrate proving capabilities with neural networks.

Table 10. Assigning the X and Y concepts to NeSy frameworks for the period 1943–1989. Meural network utilized in the NeSy framework.

ID	Reference to NeSy framework	X and Y concept	Neural network architecture
1	[79]	X1.1, X1.6.1.1.2	McCullochPitts neural network

5.2 1990–2007: Pioneering NeSy AI Era

This section surveys twelve research papers published between 1990 and 2007 that address the integration of PL and neural networks to perform reasoning or proving tasks. Assignment X and Y concepts to NeSy frameworks is given in Table 11.

The theoretical work in [91; 90] sets out a formal way to combine PL with a neural network with symmetric weights to solve the SAT problem in PL (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*; see Figure 11). The suggested framework systematically transforms any well-formed PL formula in conjunctive normal form (CNF) into a quadratic energy function. Furthermore, the author also showed that any quadratic energy function can be used to present a well-formed PL formula, thereby establishing an integration between PL and neural networks. Figure 1 in [91] demonstrates how a connectionist neural network is constructed from a well-formed PL formula. The

Table 9. Chronological grouping of NeSy frameworks for inference, reasoning and proving in PL.

Group	Years	Era	Number of papers
G1	1943–1989	Foundational neural logic	1
G2	1990–2007	Pioneering NeSy AI	12
G3	2008–2017	Foundational NeSy AI	4
G4	2018–2020	Differentiable NeSy AI	11
G5	2021–2022	Learning for reasoning	7
G6	2023–2024	Graph-based NeSy solver	5
G7	2025	Hybrid reasoning and proving	3
		TOTAL:	43

work [91] is chiefly theoretical in nature and is devoid of empirical benchmarking. For instance, it does not evaluate how well the connectionist neural network processes unsatisfiable PL formulas, nor its capacity to scale to PL formulas involving multiple propositional variables. Subsequent studies [128], have criticized the scalability and representational complexity of the symmetrical neural network approach. In a subsequent paper [92] expanded upon the work of [91] by integrating a form of PL called penalty logic [92, p. 206, Section 2] into a symmetrical neural network. It is evident that a connectionist neural network is designed to find a truth assignment that makes a PL formula true. Consequently, the framework proposed in [91] is unable to demonstrate the truth of a PL formula for all truth assignments of PL atomic propositions.

Table 11. Assigning the X and Y concepts to NeSy frameworks for the period 1990–2007. Neural networks utilized in NeSy frameworks.

ID	Reference to NeSy framework	X and Y concept	Neural network architecture
1	[91], [90]	X1.1.1.1.1	Symmetric neural network (SNN)
2	[120], [122]	X1.1	Knowledge Based Artificial Neural Network (KBANN)
3	[31], [137]	X1.6.1.1.2	Feedforward neural network (FNN)
4	[99]	X1.1.1	Feedforward neural network (FNN)
5	[59], [106]	X1.1.1	Connectionist neural network (CNN)
6	[60]	X1.1.1	Recurrent neural network (RNN)
7	[18]	X1.1.1	Hopfield neural network (HNN)
8	[130]	X1.1.1	Neural logic network (NLN)

Knowledge Based Artificial Neural Network (KBANN) [122] is one of the earliest NeSy approaches for integrating PL with trainable neural networks [120, see Abstract]. Rules expressed as Horn clauses are compiled into a neural network architecture via a *seven-step rules to network* translation process: *rewriting*, *mapping*, *numbering*, *adding hidden units*, *adding input units*, *adding links*, *perturbing* [120, Section 3.4]. The resulting network is trained using the backpropagation algorithm [120, p. 5, Section 3.1]. The KBANN performs deductive reasoning (X1.1: *Foundational reasoning*; see Table 11) based on embedded rules but does not perform automated theorem proving. The KBANN is evaluated on DNA sequence analysis [120, p. 13, Sections 4.2; p. 14, Section 4.3]. Empirical tests indicate that the KBANN are effective classifiers by comparison to other methods examined by authors [120, p. 15, Section 5].

A paper [99] shows that a feedforward neural network can be trained to compute the truth values of PL formulas (X1.1.1: *Decision mechanisms*) expressed in a disjunctive normal form (DNF). In this work, rather than encoding rules explicitly, the neural network's weights implicitly capture PL through training with the *error back-propagation* algorithm [99, Section 6]. The neural network uses a three-layer architecture with input, hidden, and output layers. A single hidden layer is sufficient for PL reasoning tasks; a feedforward neural network with three neurons in one hidden layer can learn these tasks. This approach contrasts with that of [120] in KBANN-net, which directly maps rules expressed in PL into the neural network architecture. The paper [99] provides a systematic examination of the capacity of neural networks to generalize PL reasoning; however, it does not perform theorem proving. Authors report that they tested a single hidden layer neural network to compute the truth values of PL formulas with more than two propositional letters (variables) [99]. Their results indicated that neural networks with more hidden neurons achieved lower error rates more quickly. For researchers and Ph.D. students entering the NeSy AI field, the [99] paper remains an important reference on the implicit integration strategy, as opposed to rule-based approaches that integrate PL directly into the neural network structure.

One approach in NeSy AI involves translating a PL program, expressed as general clauses [31, p. 5, Definision 1] (X1.6.1.1.2: *Logic expressivity & proof calculus support*; see Figure 11), into a feedforward neural network (FNN) [31, Section 2]. This method can translate any PL program into an FNN with one hidden layer, as it is formally proved in Theorem 3 of the same work. The translation procedure is sound [31] and allows the resulting FNN to be trained using both example data and symbolic knowledge. This enables the FNN to perform both deductive and inductive reasoning. This process is illustrated in [31, p. 12, Example 4; p. 15, Example 9], which shows how individual clauses map to neurons and connections in the FNN. The resulting FNN is depicted in [31, p. 13, Figure 3]. This approach, implemented in the connectionist system $C-IL^2P$ [46], has been applied to tasks such as DNA sequence classification. In this application, the training set, consisted of 3,190 DNA sequences, each containing 60 nucleotides, was used. The $C-IL^2P$ system performance was evaluated using cross-validation test methodology and was found to outperform the KBANN [120]. The $C-IL^2P$ system can perform inference (X1.1.1: *Decision mechanisms*) on unknown data and extract or revise propositional rules from the trained neural network [42, p. 86, Section 2.2]. Additional work integrating logic programs and connectionist neural networks is published in [59; 106]. In one integration, the authors of [59] applied the Funahashi theorem [43, p. 184, Theorem 1]. Consequently, one can construct a neural network that computes the immediate consequence operator for each PL program [106, p. 117, Theorem 2.3, Section 2.3]. Similar to the aforementioned work, the theoretical study in [60] shows that, for each PL program, there is a neural network capable of computing its semantics [60, p. 48, Section 2.1]. Using a similar approach, the PL program is integrated into a Hopfield neural network by minimizing logical inconsistencies [18].

In a research paper [137], the authors describe how they implement PL as a Boolean function (X1.6.1.1.2: *Logic expressivity & proof calculus support*; see Table 11) in a neural network. Authors use two types of neural networks: one type uses two feedforward layers, and the other uses a fully backcoupled type with N units [137, p. 181, Section 3]. This work does not discuss how reasoning and proving in PL can be performed by a neural network.

An evolutionary system that has implemented a neural logic network using genetic programming using an indirect encoding approach is proposed in [130]. The system uses PL statements as input (X1.6.1.1.2: *Logic expressivity & proof calculus support*; see Table 11) [130, p. 376, Section 4.2].

5.3 2008–2017: Foundational NeSy AI Era

The CILP++ is a system for relational learning [42]. It is a successor of the $C-IL^2P$ system. The CILP++ uses the propositionalization method called Bottom Clause Propositionalization (BCP) [42, p. 88, Section 2.3]. The BCP transforms relational data into so called propositional feature representations, yielding an attributevalue table suitable for neural-network learning. CILP++ performs neural decision-based inference (X1.1.1: *Decision mechanisms*; see Table 12) on unseen data and learns from relational data. The CILP++ system is not designed to perform proving tasks in PL.

PL programs can be encoded in a Restricted Boltzmann Machine (RBM) by translating them into the RBM's energy functions [125]. This encoding enables the RBM to carry out NeSy reasoning tasks, including solving SAT (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*; see Table 12). In a related line of work, the same author demonstrates how symbolic knowledge, expressed as if-then rules in the form of Horn clauses, can be embedded into unsupervised neural networks [126]. In this setting, reasoning is implemented directly at the neural network level. Inference over relations (X1.1.2: *Logical inference (derivation of relations)*) between entities is performed by an algorithm that operates on the trained network rather than by an external solver [126, p. 4, Algorithm 1] . This illustrates a NeSy reasoning process in which symbolic reasoning emerges as an interpretation of neural inference, rather than being performed by an explicit symbolic reasoner [126, p. 3, Section 4.2]..

Table 12. Assigning the X and Y concepts to NeSy frameworks for the period 2008–2017. Neural networks utilized in NeSy frameworks.

ID	Reference to NeSy framework	X and Y concept	Neural network architecture
1	[42]	X1.1.1	Artificial neural network (ANN) as a directed graph [42, p. 86, Section 2.2]
2	[125]	X1.1.1.1.1	Restricted Boltzmann Machine (RBM)
3	[126]	X1.1.2	Unsupervised neural network (UNN)
4	[101]	X1.6.1.1.2, Y1.3.1	Neural network for end-to-end differentiable proving

Inspired by the backward chaining algorithm, a recursive neural network architecture called Neural Theorem Prover (NTP) is constructed to perform end-to-end differentiable proving [101]. In this NeSy framework, theorems are formulated as queries to a knowledge base, and proof search is implemented through rule application and unification (Y1.3.1: *Soundness*; see Table 12). As a result, the NTP can infer a new fact from an incomplete knowledge base. While the knowledge base can be expressed in FOL, even though this is not explicitly stated in [101], it is reasonable to note that facts can also be expressed in PL (X1.6.1.1.2: *Logic expressivity & proof calculus support*; see Figure 12).

5.4 2018–2020: Differentiable Neuro-Symbolic AI Era

DeepMind suggested a way to create a dataset in the form of triples that can be used to identify logical entailment in PL [39]. Authors evaluate neural networks on this dataset and show how neural networks can capture logical entailment (X1.1.1: *Decision mechanisms*) [39]. Authors use the PossibleWorldNet neural network architecture to approximate such logical entailment. The paper compares the approach taken to other approaches and neural models [39, p. 8, Table 2].

In [7], the authors trained a neural network to solve the Circuit-SAT problem (X1.1.1.1.1:*Decision-only (SAT/UNSAT)*; see Table 13) [30, p. 1070] [129, p. 2, Section 2, Definition 3]. The input to the neural network is a PL statement, which is shown as a Boolean circuit. A proposed methodology for training the neural network involves the utilization of a neural network function to generate SAT solutions, even in the absence of exposure to SAT solutions during the training process [7, p. 6, Section 4.1]. Close to work on solving Circuit-SAT problem, the NeuroSAT model [110] introduces a message passing neural network trained as a binary classifier to predict SAT of Boolean formulas in CNF based on single bit supervision (X1.5.3: *Core UNSAT prediction and refutation*; see Table 13). These formulas are encoded as a graph with clause and literal nodes. The NeuroSAT can also be applied to solve other combinatorial problems, such as graph coloring or vertex cover problems, by encoding them as SAT problems [110, see Abstract].

Table 13. Assigning the X and Y concepts to NeSy frameworks for the period 2018–2020. Neural neworks utilized in NeSy frameworks.

ID	Reference to NeSy framework	X and Y concept	Neural network architecture
1	[39]	X1.1.1	PossibleWorldNet
2	[7]	X1.1.1.1.1, X1.5.3	×
3	[22]	X1.1.1.1.1, X1.5.3	Permutation invariant neural network (PINN)
4	[110]	X1.1.1.1.1, X1.5.3	Message passing neural network (MPNN)
5	[67]	X1.5.3, X1.3.1.2.2,	Graph neural network (GNN)
6	[141]	X1.1.1.1.1, X1.3.1.2.2	Graph neural network (GNN)
7	[134]	X1.3.2.1	SATNet model
8	[98]	X1.1.2	Logical neural network (LNN)
9	[112]	X1.1.2	Logic-integrated neural network (LINN)
10	[48]	X1.1.1.1.1	Tee neural network (TNN)
11	[132]	X1.3.1.2.2	×

The NeuroSAT solver can be applied to differential cryptanalysis [119, p. 3, Section 2.2]. In this approach, the cryptanalysis task is encoded into Boolean satisfiability problem. The NeuroSAT is then trained to classify satisfiable or unsatisfiable PL formulas.

A study on solving propositional SAT (X1.1.1.1.1:*Decision-only (SAT/UNSAT)*; see Table 13) using an end-to-end neural learning approach is presented in [22]. In this study, the SAT problem is treated as a decision problem only. It is neither an inference nor reasoning task in the PL sense. Rather than a deduction process, it can be described as a neural prediction process. The neural model is trained to predict whether a PL statement is satisfiable without producing a formal proof. PL formulas in CNF are referred to as SAT instances [22, p. 3326, Section 3] and are encoded as a matrix representation known as a clause-variable tensor. Experimental results show that the trained neural model achieved 84% prediction accuracy on 600-variable problems. The authors claim that state of the art methods take hours to solve these problems [22, see Abstract].

Graph-Q-SAT is a NeSy SAT solver trained via reinforcement learning using GNNs (X1.3.1.2.2 Reinforcement learning for clause selection) [67]. It guides Conflict Driven Clause Learning (CDCL) solver, which generates a proof of unsatisfiability [67, Introduction section, 3rd paragraph]. The SAT problem is encoded as a graph representation, similar to the approach in [110] (X1.5.3: Core UNSAT prediction and refutation; see Table 13). Graph-Q-SAT enables the trained policy to select branching variables efficiency, guiding the CDCL solver toward solutions 2-3X faster [67, p. 1, see Abstract].

The work by [141] used a GNNs to solve Boolean SAT problem (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*). This approach is different from the work by [110] which predicts SAT. The GNNs is a part of a stochastic local search (SLS) algorithm. Here, the GNNs acts as a learned variable–selection heuristic [141]. This heuristic is learned through a process called reinforcement learning (X1.3.1.2.2: *Reinforcement learning for clause selection*; see Table 13) [141, p. 5, Section 6.1]. In the experiment, the results are compared to those of WalkSAT [107] [141, p. 7, Section 7.1] by number of steps to find a solution, with a maximum of 750 steps, and by percentages of Boolean formulas that are correctly solved [141, p. 7, Table 2]. For example, if one is trying to solve a SAT problem where the maximum number of variables is 50 and the maximum number of clauses is 213 [141, p. 7, first row in Table 2], the percentage of formulas solved using the GNNs approach is higher than the percentage of formulas solved by the WalkSAT solver [141, p. 7, second and last cell in the first row of Table 2].

A differentiable, approximate MAXSAT solver, called SATNet, in proposed in [134]. SATNet implements a differentiable approximation to the maximum satisfiability (MaxSAT) problem (X1.3.2.1: *MaxSAT optimization*; see Table 13) for PL by relaxing MAXSAT problem to semidefinite program (SDP). The solver can be integrated with a larger convolutional neural network (CNN) architecture, enabling end-to-end training via gradient based learning [134, see Abstract and Section 4.3].

A logical neural network (LNN) computes truth value bounds for (sub)formulae and atoms based on initial knowledge [98, p. 5, Section 4]. This entire process is referred to as inference (X1.1.2 *Logical inference (derivation of relations)*), and it can applied to formulae in PL that are encoded as a graph structure within the LNN [98, p. 3, Section 2, Figure 1].

The tree neural network (TNN) is implemented inside the HOL proof assistant [48]. The TNN's ability to recognize pattern is evaluated on tasks, such as computing the value of an expression or estimating the truth of PL formulas (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*; see Table 13) [48, p. 1, Introduction section, 2nd paragraph].

The logic integrated neural network (LINN) is a dynamic neural network architecture that combines PL with deep learning techniques [112]. In the LINN, each propositional letter and the logical constants $\top$, $\bot$ are presented as a vector embedding, while logical connectives $\wedge$, $\vee$, $\neg$ are implemented as learnable neural modules [112, Section 3]. For given PL formula as input, the LINN dynamically builds a graph that is the mirror of the input structure. The LINN performs reasoning through this graph and produces inference output (X1.1.2: *Logical inference (derivation of relations)*) that refers to the process of computing truth value of the PL formula using trained neural model [112, see Abstract].

In [132], authors applied machine learning to automatically learn heuristics that are used to solve combinatorial problems such as SAT in PL. (X1.3.1.2.2: *Reinforcement learning for clause selection*). Authors frame the learning process in reinforcement learning settings. The method is implemented within the SAT-Gym software framework [132, see Abstract and Introduction section].

5.5 2021–2022: Learning for Reasoning Era

SimpleLogic is a problem space named by authors to study reasoning tasks in PL specified using a template consisting of facts, rules, a query, and label [144, p. 3, Section 2.1]. Visualization of a SimpleLogic problem space under two different distributions are illustrated in [144, p. 4, Figure 3]. The BERT [36] model is trained to emulate the forward-chaining algorithm (X1.1.2: *Logical inference (derivation of relations)*) [144, p. 3, Figure 2] and to solve reasoning tasks defined within SimpleLogic problem space. BERT is trained on examples drawn from a single data distribution. As a result, BERT model fails to generalize (X1.5.1: *General logical reasoning*) to other distributions over the same reasoning problem space [144, p. 5, third row in Table 1].

Table 14. Assigning the X and Y concepts to NeSy frameworks for the period 2021–2022. Neural networks utilized in NeSy frameworks.

ID	Reference to NeSy framework	X and Y concept	Neural network architecture
1	[144]	X1.1.2, X1.5.1	BERT
2	[23]	X1.1.1.1.1	Graph neural networks (GNN)
3	[40]	X1.1.1.1	Neural logic analogy learning (Noan)
4	[127]	X1.1.1.1	Restricted Boltzmann Machines (RBM)
5	[77]	X1.3.2.1	Custom supervised learning model
6	[65]	X1.1.1.1.1	Hopfield neural network (HNN)
7	[87]	X1.3.1.2.3	Recurrent neural network (RNN)

In [23], authors discuss how GNN effectively encodes combinatorial and reasoning problems. Authors describe how existing work on predicting whether a PL formula is SAT/UNSAT (X1.1.1.1.1:*Decision-only (SAT/UNSAT)*; see Table 14) using GNN, namely NeuroSAT (see 2nd paragraph in Section 5.4) and NeuroGIS can perform better than original NeuroSAT [23, p. 19, 2nd paragraph].

Neural logic analogy learning (Noan) is a neural architecture presented in [40] tha has been developed for the purpose of solving analogy problems. In this approach, the problem is converted into an evaluation problem of a PL formula. In this problem, each propositional letter is mapped to a specific letter or position in a string. The Noan neural network has been shown to learn PL letters as vector embedding and each PL operation ($\vee, \wedge, \neg$) as a neural module in order to evaluate PL formula [40, see Abstract]. Noan's approach involves utilization the PL formula within a graph structure as an input. Reasoning in Noan is implemented as a PL formula evaluation (X1.1.1.1: *Basic decisions*) and outputs a true (false) evaluation problem of the PL formula [40, see Abstract], rather than as reasoning performed by a PL reasoning engine.

Logical Boltzmann Machines (LBM) has been developed for the purpose of representing any PL formula in a neural network and performing reasoning using a probabilistic neural network referred to as Restricted Boltzmann Machines (RBM) [127, p. 2, see Introduction section] [124, p. 14, Section 2.3]. The authors present a sound algorithm that translates a PL formula into the RBM machine. The RBM assigns truth values to a PL formula (X1.1.1.1: *Basic decisions*).

In [77], authors use supervised neural learning method to approximate Boolean MAX-E-3-SAT

(X1.3.2.1: *MaxSAT optimization*; see Table 14) problem in PL. Authors encoded PL formulas using a graph structure.

Hopfield Neural Network (HNN) is trained to approximate solutions to 2SAT problem in PL (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*) [65, p. 40, Section 2.2]. Authors do not use a standard symbolic 2SAT solvers within the HNN. The 2SAT is implemented in the HNN by assigning neuron to each PL variable in PL formula expressed in CNF. In this work, approximate 2SAT is not treated as an inference, reasoning, or proving task. The HNN is used only as a tool to find the best numerical solution. It is not used as a way to formally perform inference, reasoning or proving.

QuerySAT, a recurrent neural SAT solver for solving SAT in PL with a query mechanism, is described in [87]. During the reasoning process, its query mechanism generates a candidate solution (query). This query is evaluated using unsupervised loss function that measures how well it satisfies the given SAT instance, and the evlauation result is fed back to the neural network for further interpretation (X1.3.1.2.3: *Query-aware formula generation*.

5.6 2023–2024: Graph-based NeSy AI Era

Integrating reasoning and learning in a neural network is a primary goal of NeSy AI [128]. To this end, the authors of [128] introduce a NeSy framework that can represent any PL formula [128, see Abstract]. The authors also proved the equivalence between PL and RBM [128, p. 6560, see Theorem 1]. A PL formula can be encoded in an RBM by translating it into a disjunctive normal form (DNF). The authors demonstrate that the resulting Logical Boltzmann Machine (LBM) can solve SAT problem (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*; see Table 15) in PL without training data. The LBM can not deterministically prove unsatisfiability (UNSAT) of PL formula. Additionally, the NeSy framework presented in [128] does not engage in theorem proving within PL.

Table 15. Assigning the X and Y concepts to NeSy frameworks for the period 2023–2024. Neural networks utilized in NeSy frameworks.

ID	Reference to NeSy framework	X and Y concept	Neural network architecture
1	[75]	X1.3.2.1	Graph neural network (GNN)
2	[128]	X1.1.1.1.1	Knowledge encoded networks called Logical Boltzmann Machines (LBM)
3	[135]	X1.1.1.1.1	NeuroBack: Graph neural network (GNN)
4	[143]	X1.3.2.1	Hypergraph convolutional network (HyperGCN)
5	[49]	X1.3.1.1.2, Y1.1, Y1.1.1	Graph neural network (GNN)

The GNN is trained to solve the MaxSAT problem (X1.3.2.1: *MaxSAT optimization*; see Table 15) in PL [75], demonstrating how to bridge the gap between learning and reasoning in NeSy frameworks. In PL, formulas are expressed in CNF and represented in a varibale-incidence graph structure [75, p. 1, Figure 1].

The NeuroBack NeSy framewokr was developed to overcome the limitations of directly applying GNN to predict the SAT of PL formula and to improve the effectiveness of Conflict-Driven Clause Learning (CDCL) SAT solving (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*; see Table 15) [135, p. 4, Section 4.1]. First, NeuroBack first converts a PL formula expressed in CNF into a learnable graph representation. In the subsequent inference step, a GNN predicts the likely truth values (phases) of the propositional variables. These predictions are then used to initialize and guide the CDCL solver in computing the SAT of the input PL formula. The GNN's inference enhances the efficiency of the CDCL solver's reasoning process. NeuroBack can compute UNSAT of a Pl formula that is entirely an output of the CDCL solver's reasoning process. The paper [135] does not address theorem proving tasks in PL, focusing instead on neural network inference for guiding CDCL-based SAT solving.

HyperSAT is a NeSy framework developed to predict solutions to the Weighted MaxSAT problem in PL (X1.3.2.1: *MaxSAT optimization*; see Table 15) [143]. A Weighted MaxSAT instance is formally defined as a triple consisting of a set of propositional variables, a set of propositional clauses, and a weight vector that assigns a weight to each clause [143, p. 2, Section 2.1]. HyperSAT uses hypergraph to represent the Weighted MaxSAT instance, and a neural network processes that hypergraph. However, the output of the neural network is an assignment of Boolean values to propositional variables. This is a solution to the Weighted MaxSAT problem. HeyperSAT does not perform logical reasoning or proof construction in the formal sense. It learns to predict variable assignments from data, treating MaxSAT problems as an inference task over structured input, rather than formal reasoning task in PL [143, p. 7, Figure 3, Section 4.2].

A neural resolution prover, NeuRes, is presented in [49]. Unlike models that predict the SAT in PL [135, p. 4, Section 4.1], NeuRes generates verifiable resolution proofs (Y1.1: *Verification and certification*). Its architecture uses graph-based embeddings of a PL formula expressed in CNF, processed by a message passing embedder [49, p. 12, Section 4.2]. The NeuRes model is trained via teacher forcing on expert proofs [49, p. 21, Section 5.1]. It is designed to produce easily checkable UNSAT certificates (Y1.1.1: *Certificate support*; see table 15) [49, see Abstract].

5.7 2025: Hybrid reasoning and proving Era

DeepSAT is a research framework that aims to apply recurrent neural networs to automated theorem proving in PL [6]. A PL formula is initially converted into a binary syntax tree then processed by Tree-LSTM model. This Tree-LSTM model is trained to predict whether such PL formula is tautology [6, see Abstract]. This trained model is integrated into Monte Carlo Tree Search (MCTS) agent [6, p. 6, Section 4]. DeepSAT framework constructs formal proof of validity (Y1.1.2: *Proof validation*)[6, see Abstract]

Table 16. Assigning the X and Y concepts to NeSy frameworks in 2025. Neural networks utilized in NeSy frameworks.

ID	Reference to NeSy framework	X and Y concept	Neural network architecture
1	[6]	Y1.1.2	Tree long short-term memory (Tree-LSTM) neural network
2	[83]	X1.1.1.1.1	Graph neural network (GNN)
3	[70]	X1.1.1.1	Transofmer, Long short term memory (LSTM), Graph convolution network (GCN)

A novel closest assignment supervision method is introduced to improve the training process for solving SAT in PL [83]. A graph neural network architecture is applied to predict satisfying assignments for Boolean SAT instances (X1.1.1.1.1: *Decision-only (SAT/UNSAT)*; see Table 16) in PL [83]. Both inference and reasoning terms are used throughout the paper. However, the authors do not adopt these notations in formal logic sense (see Section 2.2 and 2.3). Instead, inference is used in the machine learning sense, denoting the number of message passing iterations beyond the training distribution. The authors use reasoning term more descriptively to characterize the neural network's behavior as analogous to the continuous relaxation of MaxSAT [83, see Abstract].

The generalization behavior of transformers, graph convolution network, and long short-term memory (LSTM) neural networks when generating truth assignments ((X1.1.1.1: Basic decisions); see Table 16) for PL variables that make a PL formula true is investigated in [70]. The *PropRandom35* [70, p. 4, Section 3] data set are used to evaluate the ability of these neural architectures to truth assignments for PL variables in PL formulas that differ from training examples.

6 Evaluation of Existing Work

This section evaluates the NeSy frameworks surveyed in Section 5. Concepts from X and Y dimensions of the X–Y–Z taxonomy annotate each frameworks according to its reasoning and proving capability and assurance levels. To support systematic comparison across frameworks and publication periods, the Z dimension provides a statistical abstraction layer over the X–Y–Z taxonomy. The proposed evaluation method is general, extensible, and readily adaptable to other NeSy frameworks, taxonomies and evaluation context. The evaluation pipeline is formally defined by the following vector-valued function:

$$F(X, Y, A) = \big(F_1(X), F_2(Y), F_3(A)\big) \tag{9}$$

where individual components are defined as

$$Z1.1 = F_1(X1.a.b.c.d) = a \tag{10}$$
$$Z1.2 = F_2(Y1.a.b.c.d) = a \tag{11}$$
$$Z1.3 = F_3(A) \tag{12}$$

Here, a, b, c, and d are indices representing the position of a concept X or Y in the taxonomy hierarchy. The values $Z1.1$ and $Z1.2$ denote numerical level of reasoning and proving capability (X dimension) and assurance (Y dimension) levels, respectively as defined in the X–Y–Z taxonomy (see Section 3). A higher value of a a indicates a higher reasoning capability level (for X) or a higher proving assurance level (for Y). For example $F_1(X1.6.1.1.2) = 6$ and $F_2(Y1.1.2) = 1$. The auxiliary evaluation component $Z1.3$ is computed using the function (13)

$$F_3 : A \to \{1, 2, 3\} \tag{13}$$

where the argument A represents auxiliary metadata describing how a neural network participates in inference, reasoning or proving in PL. A level 3 denotes the strongest integration of neural learning with formal inference, reasoning or proving in PL, whereas a value 1 denotes the weakest level of integration.

The evaluation pipeline is fully automated and implemented as a Python project hosted in a GitLab repository, with the repository URL provided in Appendix A.1. Figure 8 illustrates the overall evaluation pipeline, which consists of four steps.

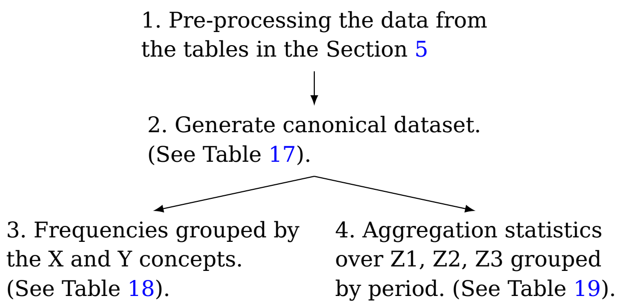

Figure 8. Automated evaluation pipeline implemented in the *nesy_evaluation* Python project (see Appendix A.1)

In the pre-processing phase (Step 1), the evaluation pipeline take as input the data presented in Tables 10 – 16 in Section 5. These data are available as CSV files within the *nesy_evaluation* Python project. Before the evaluation is executed, the table data are normalized so that each row contains exactly one feature per cell, such as a reference to NeSy framework, an X or Y concepts, or a neural network architecture name.

In Step 2, using data prepared during the pre-processing and the vector-valued function 9, the Python code computes canonical dataset shown in Table 17. During this step, numerical values for $Z1.1$, $Z1.2$, and $Z1.3$ are stored in CSV file in *data/processed* folder of the *nesy_evaluation* Python project. These data are reported in Table 17.

Table 18. Frequencies grouped by concepts from X–Y–Z taxonomy. The data shown in this table are automatically generated and stored in the *output/tables* directory of the nesy_evaluation Python project.

X and Y concept	NeSy framework count
X1.1.1.1.1	13
X1.1.1	8
X1.1.1.1	5
X1.6.1.1.2	4
X1.1.2	4
X1.5.3	4
X1.3.2.1	4
X1.1	3
X1.3.1.2.2	3
Y1.3.1	1
X1.5.1	1
X1.3.1.2.3	1
X1.3.1.1.2	1
Y1.1	1
Y1.1.1	1
Y1.1.2	1

Using the dataset generated in Step 2 and summarized in Table 17, Step 3 computes the frequencies of concepts from X–Y–Z taxonomy across the surveyed NeSy frameworks. The results are stored in the *outputs/tables* directory of the *nesy_evaluation* Python project and are reported in Table 18. As shown in Table 18, 13 NeSy frameworks support the *X1.1.1.1.1: Decision-only (SAT/UNSAT)* level of reasoning capability, whereas only one NeSy framework published during the period 2008–2017 [101] supports *Y1.3.1: Soundness* level of proving assurance. Verification and certification proving assurance are supported by two NeSy frameworks. Specifically, framework [49], published between 2023 and 2024, provides certificate support (Y1.1.1), while only the NeSy framework [6], published in 2025, supports proof validation (Y1.1.2) among 43 surveyed frameworks.

In Step 4 of the evaluation pipeline, the Python implementation computes summary statistics, namely *mean* [84, p. 190, Section 6.1, arithmetic mean definition], *median* [84, p. 223, Example 7.1; p. 200, Example 6.5], and *Shannon entropy* [111, p. 394] of the $Z1.1$, $Z1.2$, and $Z1.3$ concept within the Z evaluation space. These statistics are calculated separately for each publication period using canonical dataset summarized in Table 17. The results are stored in the *outputs/tables* directory of the *nesy_evaluation* Python project and are reported in Table 19. All numerical values in Table 19 are rounded to three decimal places. The numbers in the *count* column of Table 19 match the number of records for each publication period shown in the first column of Table 17.

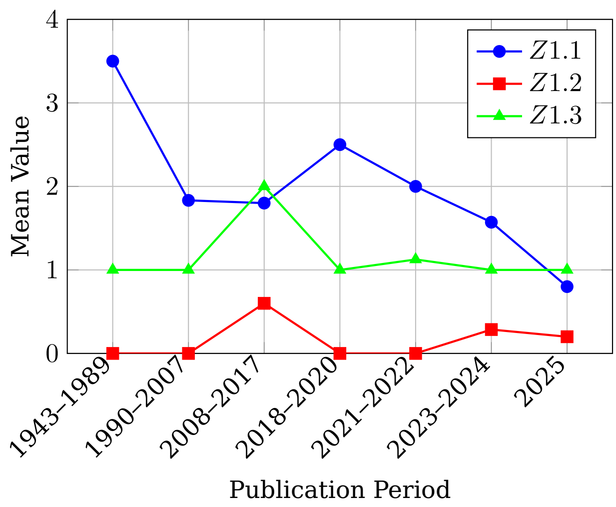

Figure 9. Mean values of the $Z1.1$, $Z1.2$, and $Z1.3$ concepts of the Z evaluation space across publication periods.

Figure 9 illustrates the mean values of the $Z1.1$, $Z1.2$, and $Z1.3$ concepts within the Z evaluation space across publication periods, as reported in Table 19. The early publication period (1943–1989) exhibited the highest mean value, 3.5, for $Z1.1$, which serves as metric for assessing the reasoning capability level of NeSy frameworks. This mean value is based on a single publication [79] from that period. Subsequent to 2020, the mean for $Z1.1$ generally exhibits downward trend, reaching a nadir of 0.8 in 2025. This finding suggests that earlier literature on NeSy AI placed greater emphasis on stronger reasoning capability level.

Table 17. Canonical NeSy evaluation dataset derived from annotated NeSy frameworks surveyed in Section 5.

ID	Reference to NeSy framework	X and Y concept	Neural network architecture name	Publication period	Z1	Z2	Z3
1	[79]	X1.1	McCulloch-Pitts neural network	1943–1989	1	0	1
2	[79]	X1.6.1.1.2	McCulloch-Pitts neural network	1943–1989	6	0	1
1	[91]	X1.1.1.1.1	Symmetric neural network (SNN)	1990–2007	1	0	1
2	[90]	X1.1.1.1.1	Symmetric neural network (SNN)	1990–2007	1	0	1
3	[120]	X1.1	Knowledge Based Artificial Neural Network (KBANN)	1990–2007	1	0	1
4	[122]	X1.1	Knowledge Based Artificial Neural Network (KBANN)	1990–2007	1	0	1
5	[31]	X1.6.1.1.2	Feedforward neural network (FNN)	1990–2007	6	0	1
6	[137]	X1.6.1.1.2	Feedforward neural network (FNN)	1990–2007	6	0	1
7	[99]	X1.1.1	Feedforward neural network (FNN)	1990–2007	1	0	1
8	[59]	X1.1.1	Connectionist neural network (CNN)	1990–2007	1	0	1
9	[106]	X1.1.1	Connectionist neural network (CNN)	1990–2007	1	0	1
10	[60]	X1.1.1	Recurrent neural network (RNN)	1990–2007	1	0	1
11	[18]	X1.1.1	Hopfield neural network (HNN)	1990–2007	1	0	1
12	[130]	X1.1.1	Neural logic network (NLN)	1990–2007	1	0	1
1	[42]	X1.1.1	Artificial neural network (ANN) as a directed graph	2008–2017	1	0	1
2	[125]	X1.1.1.1.1	Restricted Boltzmann Machine (RBM)	2008–2017	1	0	2
3	[126]	X1.1.2	Unsupervised neural network (UNN)	2008–2017	1	0	1
4	[101]	X1.6.1.1.2	Neural network for end-to-end differentiable proving	2008–2017	6	0	3
5	[101]	Y1.3.1	Neural network for end-to-end differentiable proving	2008–2017	0	3	3
1	[39]	X1.1.1	PossibleWorldNet	2018–2020	1	0	1
2	[7]	X1.1.1.1.1	×	2018–2020	1	0	1
3	[7]	X1.5.3	×	2018–2020	5	0	1
4	[22]	X1.1.1.1.1	Permutation invariant neural network (PINN)	2018–2020	1	0	1
5	[22]	X1.5.3	Permutation invariant neural network (PINN)	2018–2020	5	0	1
6	[110]	X1.1.1.1.1	Message passing neural network (MPNN)	2018–2020	1	0	1
7	[110]	X1.5.3	Message passing neural network (MPNN)	2018–2020	5	0	1
8	[67]	X1.5.3	Graph neural network (GNN)	2018–2020	5	0	1
9	[67]	X1.3.1.2.2	Graph neural network (GNN)	2018–2020	3	0	1
10	[141]	X1.1.1.1.1	Graph neural network (GNN)	2018–2020	1	0	1
11	[141]	X1.3.1.2.2	Graph neural network (GNN)	2018–2020	3	0	1
12	[134]	X1.3.2.1	SATNet model	2018–2020	3	0	1
13	[98]	X1.1.2	Logical neural network (LNN)	2018–2020	1	0	1
14	[112]	X1.1.2	Logic-integrated neural network (LINN)	2018–2020	1	0	1
15	[48]	X1.1.1.1.1	Tee neural network (TNN)	2018–2020	1	0	1
16	[132]	X1.3.1.2.2	×	2018–2020	3	0	1
1	[144]	X1.1.2	BERT	2021–2022	1	0	1
2	[144]	X1.5.1	BERT	2021–2022	5	0	1
3	[23]	X1.1.1.1.1	Graph neural networks (GNN)	2021–2022	1	0	1
4	[40]	X1.1.1.1	Neural logic analogy learning (Noan)	2021–2022	1	0	1
5	[127]	X1.1.1.1	Restricted Boltzmann Machines (RBM)	2021–2022	1	0	2
6	[77]	X1.3.2.1	Custom supervised learning model	2021–2022	3	0	1
7	[65]	X1.1.1.1.1	Hopfield neural network (HNN)	2021–2022	1	0	1
8	[87]	X1.3.1.2.3	Recurrent neural network (RNN)	2021–2022	3	0	1
1	[75]	X1.3.2.1	Graph neural network (GNN)	2023–2024	3	0	1
2	[128]	X1.1.1.1.1	Logical Boltzmann Machines (LBMs)	2023–2024	1	0	1
3	[135]	X1.1.1.1.1	NeuroBack: Graph neural network (GNN)	2023–2024	1	0	1
4	[143]	X1.3.2.1	Hypergraph convolutional network (HyperGCN)	2023–2024	3	0	1
5	[49]	X1.3.1.1.2	Graph neural network (GNN)	2023–2024	3	0	1
6	[49]	Y1.1	Graph neural network (GNN)	2023–2024	0	1	1
7	[49]	Y1.1.1	Graph neural network (GNN)	2023–2024	0	1	1
1	[6]	Y1.1.2	Tree long short-term memory (Tree-LSTM) neural network	2025	0	1	1
2	[83]	X1.1.1.1.1	Graph neural network (GNN)	2025	1	0	1
3	[70]	X1.1.1.1	Transformer	2025	1	0	1
4	[70]	X1.1.1.1	Long short term memory (LSTM)	2025	1	0	1
5	[70]	X1.1.1.1	Graph convolution network (GCN)	2025	1	0	1

Table 19. Mean, median and entropy computed for $Z1.1$, $Z1.2$, and $Z1.3$, including number of publications given in Count column. Data are grouped by publication period based on data summarized in Table 9.

Publication period	$Z1.1$ mean	$Z1.1$ median	$Z1.1$ entropy	$Z1.2$ mean	$Z1.2$ median	$Z1.2$ entropy	$Z1.3$ mean	$Z1.3$ median	$Z1.3$ entropy	count
1943–1989	3.5	3.5	1.0	0.0	0.0	0.0	1.0	1.0	0.0	2
1990–2007	1.833	1.0	0.650	0.0	0.0	0.0	1.0	1.0	0.0	12
2008–2017	1.8	1.0	1.371	0.6	0.0	0.722	2.0	2.0	1.522	5
2018–2020	2.5	2.0	1.5	0.0	0.0	0.0	1.0	1.0	0.0	16
2021–2022	2.0	1.0	1.299	0.0	0.0	0.0	1.125	1.0	0.544	8
2023–2024	1.571	1.0	1.557	0.286	0.0	0.863	1.0	1.0	0.0	7
2025	0.8	1.0	0.722	0.2	0.0	0.722	1.0	1.0	0.0	5

In contrast, more recent publications have demonstrated a wider range of reasoning capability levels. Conversely, the results presented in Table 18 indicate that concepts from the Y dimension, which are design to capture the proving assurance levels, are present in only four instances among the 43 publications that were surveyed. The mean values for $Z1.2$, which reflects the numerical representation of concepts from Y dimension, as illustrated in Figure 9, are generally zero or very low. This observation suggests that the proving assurance level is either rarely considered in the extant literature or is a recently introduced concept.

Overall, the mean values of $Z1.3$ demonstrate relative stability across publication periods, maintaining proximity to 1.0 in the majority of cases. A mean value of 2.0 is observed in the publication period 2008–2017. This finding suggests that the integration of neural learning with formal inference, reasoning or proving in PL is a persistent theme in the extant literature.

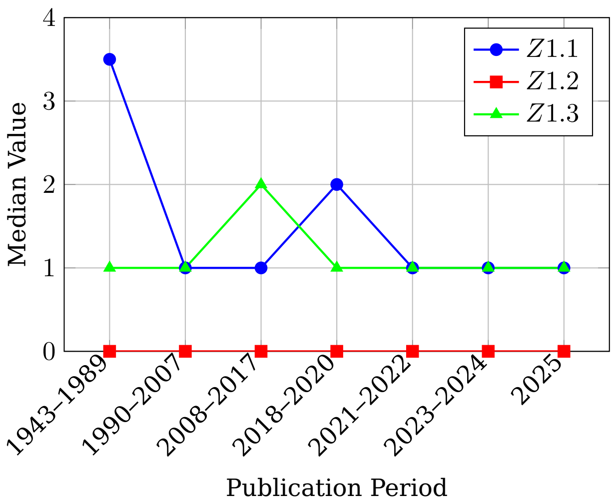

Figure 10. Median values of the $Z1.1$, $Z1.2$, and $Z1.3$ concepts of the Z evaluation space across publication periods.

Figure 10 illustrates the median values of the $Z1.1$, $Z1.2$, and $Z1.3$ concepts of the Z evaluation space are depicted across various publication periods, as reported in the Table 19. The median values of $Z1.1$ captures the typical (central) level of reasoning capability observed among NeSy frameworks in a given publication period. The median of $Z1.1$ exhibited the highest value in the publication period from 1943 to 1989. Since 1990, the median value has remained steady, except for the period from 2018 to 2020, during which it reached to 2.0. This pattern indicates that, during the most active publication period, the majority of NeSy frameworks depend on a particular level of reasoning capability. Conversely, the median value of the $Z1.2$ concept within the Z evaluation space is observed to be zero across all publication periods, as illustrated in Figure 10. This finding suggests the absence of assurance mechanisms in the majority of the NeSy frameworks. Concurrently, the median value of the $Z1.3$ within the Z evaluation space is equivalent to 1 in the majority of cases, with the exception of the pick it reaches of 2.0 for the period between 2018 and 2020. Nevertheless, this configuration results in a similar conclusion regarding the integration of neural learning with formal inference, reasoning, or proving in PL as evidenced by the mean value of the $Z1.3$.

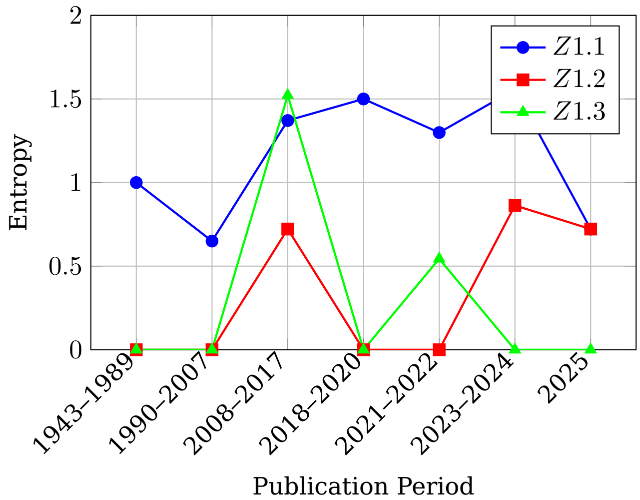

Figure 11. Entropy values of the $Z1.1$, $Z1.2$, and $Z1.3$ concepts of the Z evaluation space across publication periods.

Figure 11 illustrates the Shannon entropy of the $Z1.1$, $Z1.2$, and $Z1.3$ concepts of the Z evaluation space varies across publication periods. Notably,

$Z1.1$ exhibits an Shannon entropy value greater than or equal to 0.650 throughout all periods. The data show a consistent upward trend from 1990 to 2024, reflecting a more even distribution of reasoning capability levels, followed by a slight decline to 0.722 in 2025. Overall, the Shannon entropy values for $Z1.1$ indicate reasoning capability levels in PL are distributed more evenly across NeSy frameworks over the period of time (1990–2024).

However, the Shannon entropy values for $Z1.2$ remain zero from 1943 to 2022, except for a value of 0.722 during the 20082017 period. The entropy then increases before decreasing again to 0.722 in 2025. Overall, this pattern indicates that a single proving assurance level (Y1.1) dominates across most publication periods(see frequencies in Table 18), indicating minimal variation in how NeSy frameworks provide proving assurance.

The Shannon entropy values for $Z1.3$ remain zero across nearly all publication periods, except for 2008-2017, when it reached 1.522, and for 2021-2022, when it rose to 0.544. These low entropy values indicate that the integration of neural learning with formal inference, reasoning, or proving in PL has not been prevalent in the NeSy literature over time.

7 Conclusions

In this survey, the landscape of NeSy frameworks with respect to their level of capability and assurance to perform reasoning and proving tasks in PL is examined. To enable structured comparison and evaluation, we introduced a novel X–Y–Z taxonomy specifically tailored to NeSy frameworks that address any level of reasoning and proving capability or assurance in PL. This X–Y–Z taxonomy has been formalized as an ontology following the SKOS standard. We clarified the boundaries and overlaps between inference, reasoning, and proving in PL, highlighting their interchangeable use in the context of NeSy AI.

The proposed X–Y–Z taxonomy is applied to survey and evaluate existing NeSy frameworks, revealing strengths, limitations, and gaps in current approaches to reasoning and proving capability or assurance level in PL. Additionally, we applied the XYZ taxonomy to analyze existing survey papers on NeSy AI that discuss the NeSy reasoning or proving tasks in PL. This allowed us to align the taxonomies employed in those surveys with our XYZ framework, providing a unified perspective on how reasoning and proving in PL are addressed across the literature.

A fully automated, reproducible evaluation pipeline in Python is implemented by utilizing X–Y–Z taxonomy. The implementation computes aggregate statistics (mean, median, and entropy) over the surveyed frameworks and is publicly available on GitHub, allowing other researchers to replicate, extend, or update the evaluation with new frameworks or metrics.

Overall, by offering a taxonomy as a classification tool, and an open, executable evaluation technique centered on PL reasoning and proving, this work promotes the maturity of the NeSy discipline. These contributions, in our opinion, make direct comparisons easier, and lead the development of future NeSy systems that more effectively strike a balance between capability and assurance levels in MeSy reasoning and proving.

A Appendix

A.1 Supplementary Material

1. The *nesy_evaluation* Python project: https://gitlab.com/TIBHannover/human-decision/neuro-symbolic-reasoning-survey/-/tree/main/nesy_evaluation
2. SPARQL query: https://gitlab.com/TIBHannover/human-decision/neuro-symbolic-reasoning-survey/-/tree/main/sparql
3. Implemented X–Y–Z taxonomy as instance of SKOS ontology: https://gitlab.com/TIBHannover/human-decision/neuro-symbolic-reasoning-survey/-/tree/main/taxonomy

B Acknowledgements

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)–Projektnummer 543081196 (HUMAN-DECISION).

During the preparation of this manuscript, the authors used ChatGPT (OpenAI) and DeepL tools to improve the readability of the text and to fix grammatical errors. ChatGPT (OpenAI) is used to build a taxonomy as instance of SKOS ontology, and to generate an initial version of Python code used as part of the evaluation workflow. The authors reviewed, adapted, and validated all the generated material. The authors are responsible for the scientific content, the accuracy of the ontology design and implementation, and the results reported in this paper.

The authors would like to thank reviewers for their valuable comments and suggestions.

References

[1] Acharya K, Raza W, Dourado C, Velasquez A and Song HH (2023) Neurosymbolic reinforcement learning and planning: A survey. *IEEE Transactions on Artificial Intelligence* 5(5): 1939–1953. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10238788.

[2] Ahmed K, Wang E, Chang KW and Van den Broeck G (2022) Neuro-symbolic entropy regularization. In:

Proceedings of the 38th Conference on Uncertainty in Artificial Intelligence (UAI), Proceedings of Machine Learning Research, volume 180. PMLR, pp. 43–53. URL https://proceedings.mlr.press/v180/ahmed22a.html.

[3] Akter MS, Sultan MF and Cuzzocrea A (2025) Neuro-symbolic methods in natural language processing: A review. In: *In Proceedings of the 14th International Conference on Data Science, Technology and Applications (DATA 2025), pages 274-282*. URL https://www.scitepress.org/Papers/2025/134531/134531.pdf.

[4] Al-Odhari AM (2021) Features of propositional logic. *Pure Mathematical Sciences* 10(1): 35–44. DOI:10.12988/pms.2021.91275. URL https://www.m-hikari.com/pms/pms-2021/pms-1-2021/p/alodhariPMS1-2021.pdf.

[5] Alshiekh M, Bloem R, Ehlers R, Könighofer B, Niekum S and Topcu U (2018) Safe reinforcement learning via shielding. In: *Proceedings of the AAAI conference on artificial intelligence*, volume 32. URL https://ojs.aaai.org/index.php/AAAI/article/view/11797/11656.

[6] Amato G, Balestra N, Maggesi M, Parton M et al. (2025) Recurrent neural networks for guiding proof search in propositional logic. In: *Proceedings of the 26th Italian Conference on Theoretical Computer Science*, volume 4039. CEUR-WS, pp. 113–126. URL https://ceur-ws.org/Vol-4039/paper18.pdf.

[7] Amizadeh S, Matusevych S and Weimer M (2018) Learning to solve circuit-sat: An unsupervised differentiable approach. In: *International conference on learning representations*. URL https://openreview.net/pdf?id=BJxgz2R9t7.

[8] Avigad J (2022) *Semantics of Propositional Logic*. Cambridge University Press, pp. 59–82. DOI:https://doi.org/10.1017/9781108778756.004.

[9] Awolesi AO (2025) Neuro-symbolic generative AI for explainable reasoning. *International Journal of Science and Research Archive* 16(01): 121–125. DOI:10.30574/ijsra.2025.16.1.2019. URL https://doi.org/10.30574/ijsra.2025.16.1.2019.

[10] Bachmair L and Ganzinger H (2001) Resolution theorem proving. *Handbook of automated reasoning* 1(02). URL https://lawrencecpaulson.github.io/papers/bachmair-hbar-resolution.pdf.

[11] Bader S and Hitzler P (2005) Dimensions of neural-symbolic integration-a structured survey. *arXiv preprint cs/0511042* URL https://arxiv.org/pdf/cs/0511042.

[12] Balendran A, Beji C, Bouvier F and et al (2025) A scoping review of robustness concepts for machine learning in healthcare. *npj Digital Medicine* 8: 38. DOI:10.1038/s41746-024-01420-1. URL https://doi.org/10.1038/s41746-024-01420-1.

[13] Barwise J (1977) An introduction to first-order logic. In: *Studies in Logic and the Foundations of Mathematics*, volume 90. Elsevier, pp. 5–46. URL https://www.sciencedirect.com/science/chapter/bookseries/pii/S0049237X08710978.

[14] Belle V (2017) Weighted model counting with function symbols. In: *Conference on Uncertainty in Artificial Intelligence*.

[15] Berlot-Attwell I (2021) Neuro-symbolic VQA: A review from the perspective of AGI desiderata. *CoRR* abs/2104.06365. URL https://arxiv.org/abs/2104.06365.

[16] Besold TR, dAvila Garcez A, Bader S, Bowman H, Domingos P, Hitzler P, Kühnberger K, Lamb LC, Lima PMV, de Penning L, Pinkas G, Poon H and Zaverucha G (2021) Neuralsymbolic learning and reasoning: A survey and interpretation. In: *Neuro-Symbolic Artificial Intelligence: The State of the Art, Frontiers in Artificial Intelligence and Applications*, volume 342. IO Press, pp. 1–51. URL https://ebooks.iospress.nl/volumearticle/58841. Arxiv: https://arxiv.org/pdf/1711.03902.

[17] Bhuyan BP, Ramdane-Cherif A, Tomar R and Singh TP (2024) Neuro-symbolic artificial intelligence: a survey. *Neural Computing & Applications* 36(21): 12809–12844. DOI: 10.1007/s00521-024-09960-z. URL https://doi.org/10.1007/s00521-024-09960-z.

[18] bin Wan Abdullah WAT (1996) Logic programming in neural networks. *Malaysian Journal of Computer Science* 9(1): 1–5. URL https://jummec.um.edu.my/index.php/MJCS/article/view/2888/1060.

[19] Bizzarri A, Yu CEJ, Jalaian B, Riguzzi F and Bastian ND (2025) Neurosymbolic ai for network intrusion detection systems: A survey. *Journal of information security and applications* 94: 104205. URL https://www.sciencedirect.com/science/article/pii/S221421262500242X.

[20] Bonet ML, Levy J and Manyà F (2007) Resolution for max-sat. *Artificial Intelligence* 171(8): 606–618. DOI:https://doi.org/10.1016/j.artint.2007.03.001. URL https://www.sciencedirect.com/science/article/pii/S0004370207000422.

[21] Bouneffouf D and Aggarwal CC (2022) Survey on applications of neurosymbolic artificial intelligence. *arXiv preprint arXiv:2209.12618* URL https://arxiv.org/pdf/2209.12618.

[22] Cameron C, Chen R, Hartford J and Leyton-Brown K (2020) Predicting propositional satisfiability via end-to-end learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34. pp. 3324–3331. URL https://aaai.org/ojs/index.php/AAAI/article/view/5733/5589.

[23] Cappart Q, Chételat D, Khalil EB, Lodi A, Morris C and Veličković P (2023) Combinatorial optimization and reasoning with graph neural networks. *Journal of Machine Learning Research* 24(130): 1–61. URL https://www.jmlr.org/papers/volume24/21-0449/21-0449.pdf.

[24] Chanin D and Hunter A (2023) Neuro-symbolic commonsense social reasoning. URL https://arxiv.org/abs/2303.08264.

[25] Chavira M and Darwiche A (2008) On probabilistic inference by weighted model counting. *Artificial Intelligence* 172(6): 772–799. DOI: https://doi.org/10.1016/j.artint.2007.11.002. URL https://www.sciencedirect.com/science/article/pii/S0004370207001889.

[26] Chen H, Li Y, Shi S, Liu S, Zhu H and Zhang Y (2022) Graph collaborative reasoning. In: *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. pp. 75–84.

[27] Chen MK (2025) A comparative study of neurosymbolic ai approaches to interpretable logical reasoning. In: Gilpin H, Giunchiglia L, Hitzler P and van Krieken E (eds.) *Proceedings of The 19th International Conference on Neurosymbolic Learning and Reasoning*, *Proceedings of Machine Learning Research*, volume 284. PMLR, pp. 447–462. URL https://proceedings.mlr.press/v284/chen25b.html.

[28] Cheng K, Ahmed NK, Rossi RA, Willke T and Sun Y (2025) Neural-symbolic methods for knowledge graph reasoning: A survey. *ACM Transactions on Knowledge Discovery from Data* 18(9): 1–44. URL https://dl.acm.org/doi/pdf/10.1145/3686806.

[29] Colelough BC and Regli W (2025) Neuro-symbolic ai in 2024: A systematic review. *arXiv preprint arXiv:2501.05435* URL https://arxiv.org/pdf/2501.05435.

[30] Cormen TH, Leiserson CE, Rivest RL and Stein C (2009) *Introduction to Algorithms*. 3rd edition. The MIT Press. ISBN 9780262033848.

[31] d'Avila Garcez AS, Broda K and Gabbay DM (1999) Symbolic knowledge extraction from trained neural networks: The cilp system. *Applied Intelligence* 9(1): 13–33. DOI:10.1023/A:1008325906203. URL https://www.staff.city.ac.uk/~aag/papers/cilp.pdf.

[32] d'Avila Garcez AS, Broda K and Gabbay DM (2012) Neural-symbolic learning systems - foundations and applications. In: *Perspectives in Neural Computing*. URL https://api.semanticscholar.org/CorpusID:60656463.

[33] de Moura L, Kong S, Avigad J, van Doorn F and von Raumer J (2015) The lean theorem prover (system description). In: *Automated Deduction – CADE-25*, *Lecture Notes in Computer Science*, volume 9195. Springer, pp. 378–388. DOI:10.1007/978-3-319-21401-6_26.

[34] Delong LN, Mir RF and Fleuriot JD (2025) Neurosymbolic AI for reasoning over knowledge graphs: A survey. *IEEE Trans. Neural Networks Learn. Syst.* 36(5): 7822–7842. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10603423.

[35] Delvecchio GP, Molfetta L and Moro G (2025) Neuro-symbolic artificial intelligence: A task-directed survey in the black-box models era. In: Kwok J (ed.) *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*. International Joint Conferences on Artificial Intelligence Organization, pp. 10418–10426. DOI: 10.24963/ijcai.2025/1157. URL https://doi.org/10.24963/ijcai.2025/1157.

[36] Devlin J, Chang MW, Lee K and Toutanova K (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. In: Burstein J, Doran C and Solorio T (eds.) *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI:10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

[37] Enderton HB (2001) *A mathematical introduction to logic*. Elsevier. URL https://shop.elsevier.com/books/a-mathematical-introduction-to-logic/enderton/978-0-08-049646-7.

[38] Evans R and Grefenstette E (2018) Learning explanatory rules from noisy data. *Journal of Artificial Intelligence Research* 61: 1–64. URL https://www.jair.org/index.php/jair/article/download/11172/26376/.

[39] Evans R, Saxton D, Amos D, Kohli P and Grefenstette E (2018) Can neural networks understand logical entailment? URL https://arxiv.org/abs/1802.08535.

[40] Fan Y and Zhang Y (2022) Neural logic analogy learning. In: *Proceedings of the PAIR2Struct Workshop at ICLR 2022*. URL https://openreview.net/forum?id=rGe-qsNrx5.

[41] Fitas R (2025) Neuro-symbolic ai for advanced signal and image processing: A review of recent trends and future directions. *IEEE Access* URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=11124915.

[42] França MV, Zaverucha G and Garcez ASd (2014) Fast relational learning using bottom clause propositionalization with artificial neural networks. *Machine learning* 94(1): 81–104. URL https://link.springer.com/content/pdf/10.1007/s10994-013-5392-1.pdf.

[43] Funahashi KI (1989) On the approximate realization of continuous mappings by neural networks. *Neural networks* 2(3): 183–192. URL https://math.bu.edu/people/mkon/MA751/FunahashiTheorem.pdf.

[44] Garcez Ad, Gori M, Lamb LC, Serafini L, Spranger M and Tran SN (2019) Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *Journal of Applied Logics* 6(4): 611–631. URL https://arxiv.org/pdf/1905.06088.

[45] Garcez ASd, Gabbay DM, Ray O and Woods J (2007) Abductive reasoning in neural-symbolic systems. *Topoi* 26(1): 37–49. URL https://link.springer.com/content/pdf/10.1007/s11245-006-9005-5.pdf.

[46] Garcez ASd and Zaverucha G (1999) The connectionist inductive learning and logic programming system. *Applied Intelligence* 11(1): 59–77. URL https://link.springer.com/content/pdf/10.1023/A:1008328630915.pdf.

[47] Garey MR and Johnson DS (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman and Company. URL https://perso.limos.fr/~palafour/PAPERS/PDF/Garey-Johnson79.pdf.

[48] Gauthier T (2020) Tree neural networks in hol4 URL https://arxiv.org/abs/2009.01827.

[49] Ghanem MA (2024) Neures: Neural reasoning with certificates. *Journal of Machine Learning Research* 25(1): 1–35. URL https://publications.cispa.de/articles/thesis/NeuRes_A_Neural_Resolution_Prover_of_Unsatisfiability/27118017/1/files/49444704.pdf.

[50] Gibaut W, Pereira L, Grassiotto F, Osorio A, Gadioli E, Munoz A, Gomes S and Santos Cd (2023) Neurosymbolic ai and its taxonomy: a survey. *arXiv preprint arXiv:2305.08876* DOI:10.48550/arXiv.2305.08876.

[51] Gordeev L and Haeusler EH (2019) Proof compression and np versus pspace. *Studia Logica* 107(1): 53–83. URL https://link.springer.com/article/10.1007/s11225-017-9773-5.

[52] Gorré R (2006?) An introduction to classical propositional logic: Syntax, semantics, sequents. In: Lauer H (ed.) *Foundations of the Pure Sciences, in History and Philosophy of Science for Science Students Committed to Africa*, chapter 49. Ghana University Press, Lagon, Ghana, pp. 3–45. URL https://www.cs.brandeis.edu/~cs112/2006-cs112/docs/intrologic1.pdf.

[53] Gurney K (1997) *An Introduction to Neural Networks*. London and New York: University of Sheffield.

[54] Hamilton K, Nayak A, Boi B and Longo L (2022) Is neuro-symbolic ai meeting its promises in natural language processing? a structured review. *Semantic Web Interoperability, Usability, Applicability* URL https://www.semantic-web-journal.net/system/files/swj3228.pdf.

[55] Harrison J (2009) *Handbook of Practical Logic and Automated Reasoning*. Cambridge University Press. ISBN 9780521899574. URL https://kwarc.info/teaching/sWuV/harrison_handbook-of-practical-logic.pdf.

[56] Hatzilygeroudis I and Prentzas J (2000) Neurules: improving the performance of symbolic rules. *International Journal on Artificial Intelligence Tools* 9(01): 113–130. URL https://www.worldscientific.com/doi/abs/10.1142/S0218213000000094.

[57] Heule MJ, Hunt WA and Wetzler N (2013) Trimming while checking clausal proofs. In: *2013 Formal Methods in Computer-Aided Design*. pp. 181–188. DOI:10.1109/FMCAD.2013.6679408.

[58] Heule MJH, Kiesl B, Seidl M and Biere A (2017) Pruning through satisfaction. In: *Haifa Verification Conference (HVC)*. Springer, pp. 179–194. DOI:10.1007/978-3-319-67205-8_13. URL https://www.cs.utexas.edu/~marijn/publications/PRune.pdf.

[59] Hitzler P, Hölldobler S and Seda AK (2004) Logic programs and connectionist networks. *Journal of Applied Logic* 2(3): 245–272. DOI: https://doi.org/10.1016/j.jal.2004.03.002. URL https://www.sciencedirect.com/science/article/pii/S1570868304000151. Neural-symbolic Systems.

[60] Hölldobler S and Kalinke Y (1999) Approximating the semantics of logic programs by recurrent neural networks. *Applied Intelligence* 11(1): 45–58. URL https://link.springer.com/content/pdf/10.1023/A:1008376514077.pdf.

[61] Huang J, Li Z, Naik M and Lim SN (2025) Laser: A neuro-symbolic framework for learning spatial-temporal scene graphs with weak supervision. URL https://arxiv.org/abs/2304.07647.

[62] Jaakkola R, Janhunen T, Kuusisto A, Rankooh MF and Vilander M (2025) Explainability via short formulas: the case of propositional logic with implementation. *Journal of Artificial Intelligence Research* 83. URL https://www.jair.org/index.php/jair/article/view/17422.

[63] Jarmuek T (2020) *Tableau Methods for Propositional Logic and Term Logic, Studies in Philosophy, History of Ideas and Modern Societies*, volume 20. Berlin, Bern, Bruxelles, New York, Oxford, Warszawa, Wien: Peter Lang GmbH, Internationaler Verlag der Wissenschaften. ISBN 9783631833728. URL https://kwarc.info/teaching/sWuV/harrison_handbook-of-practical-logic.pdf.

[64] Kautz H (2022) The third ai summer: Aaai robert s. engelmore memorial lecture. *Ai magazine* 43(1): 105–125. DOI:https://doi.org/10.1002/aaai.12036. URL https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/19122.

[65] Kho LC, Kasihmuddin MS, Mansor MA and Sathasivam S (2022) Propositional satisfiability logic via ant colony optimization in hopfield neural network. *Malaysian Journal of Mathematical Sciences* 16(1). URL http://119.40.117.179/fullpaper/2022-January-16-1/Kasihmuddin,%20M.%20S.%20M.-37-53.pdf.

[66] Kraetzschmar G, Sablatnög S, Enderle S and Palm G (1998) Application of neurosymbolic integration

for environment modelling in mobile robots. In: *International Workshop on Hybrid Neural Systems*. Springer, pp. 387–401. URL https://link.springer.com/chapter/10.1007/10719871_27.

[67] Kurin V, Godil S, Whiteson S and Catanzaro B (2020) Can q-learning with graph networks learn a generalizable branching heuristic for a sat solver? *Advances in Neural Information Processing Systems* 33: 9608–9621. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6d70cb65d15211726dcce4c0e971e21c-Paper.pdf.

[68] Lamb LC, Garcez A, Gori M, Prates M, Avelar P and Vardi M (2020) Graph neural networks meet neural-symbolic computing: A survey and perspective. *arXiv preprint arXiv:2003.00330* URL https://arxiv.org/pdf/2003.00330.

[69] Lammich P (2020) Efficient verified (un) sat certificate checking. *Journal of Automated Reasoning* 64(3): 513–532. URL https://doi.org/10.1007/s10817-019-09525-z.

[70] Langedijk A, Jumelet J and Zuidema W (2025) Propositional logic for probing generalization in neural networks. *arXiv preprint arXiv:2506.08978* URL https://arxiv.org/pdf/2506.08978?

[71] Li V, Fu Y, Knappe T, Han K and Zhu K (2025) Automating mathematical proof generation using large language model agents and knowledge graphs. *arXiv preprint* abs/2503.11657. URL https://arxiv.org/abs/2503.11657.

[72] Li Z, Sun J, Murphy L, Su Q, Li Z, Zhang X, Yang K and Si X (2024) A survey on deep learning for theorem proving. *arXiv preprint arXiv:2404.09939* URL https://arxiv.org/pdf/2404.09939.

[73] Liang B, Wang Y and Tong C (2025) Ai reasoning in deep learning era: From symbolic ai to neural–symbolic ai. *Mathematics* 13(11): 1707. URL https://www.mdpi.com/2227-7390/13/11/1707.

[74] Liberatore P (2008) Redundancy in logic ii: 2cnf and horn propositional formulae. *Artificial Intelligence* 172(2-3): 265–299. DOI:10.1016/j.artint.2007.06.003. URL https://www.sciencedirect.com/science/article/pii/S0004370207001105.

[75] Liu M, Huang P, Jia F, Zhang F, Sun Y, Cai S, Ma F and Zhang J (2023) Can graph neural networks learn to solve maxsat problem? *Proceedings of the AAAI Conference on Artificial Intelligence* 37(13): 16264–16265. URL https://ojs.aaai.org/index.php/AAAI/article/view/26992.

[76] Manhaeve R, Dumancic S, Kimmig A, Demeester T and De Raedt L (2018) Deepproblog: Neural probabilistic logic programming. *Advances in neural information processing systems* 31. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/dc5d637ed5e62c36ecb73b654b05ba2a-Paper.pdf.

[77] Marino R (2021) Learning from survey propagation: A neural network for max-e-3-sat. *Machine Learning: Science and Technology* 2(3): 035032. URL https://iopscience.iop.org/article/10.1088/2632-2153/ac0496/pdf.

[78] Marra G, Dumani S, Manhaeve R and De Raedt L (2024) From statistical relational to neurosymbolic artificial intelligence: A survey. *Artificial Intelligence* 328: 104062. DOI:https://doi.org/10.1016/j.artint.2023.104062. URL https://www.sciencedirect.com/science/article/pii/S0004370223002084.

[79] McCulloch WS and Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics* 5(4): 115–133. URL https://jontalle.web.engr.illinois.edu/uploads/410-NS.F22/McCulloch-Pitts-1943-neural-networks-ocr.pdf.

[80] McCune W (2005–2010) Prover9 and mace4. URL https://github.com/ai4reason/Prover9.

[81] McGuinness L (2021) Prisma 2020 checklist shiny user-friendly interface (app). URL https://prisma.shinyapps.io/checklist/. Userfriendly interface for the PRISMA 2020 checklist. Accessed 12 Nov 2025.

[82] Miles A and Bechhofer S (2009) Skos simple knowledge organization system reference. URL https://www.w3.org/2009/08/skos-reference/skos.html. W3C Recommendation, 18 August 2009.

[83] Mojzisek D, Hula J, Li Z, Zhou Z and Janota M (2025) Neural approaches to sat solving: Design choices and interpretability. *arXiv preprint arXiv:2504.01173* URL https://arxiv.org/pdf/2504.01173?

[84] Montgomery DC and Runger GC (2014) *Applied Statistics and Probability for Engineers*. 6th edition. Hoboken, NJ, USA: Wiley. See chapter on measures of central tendency for sample mean and median definitions.

[85] Nawaz U, ur Rahaman MA and Saeed Z (2025) A review of neuro-symbolic ai integrating reasoning and learning for advanced cognitive systems. *Intelligent Systems with Applications* 26: 200541. DOI:https://doi.org/10.1016/j.iswa.2025.200541. URL https://www.sciencedirect.com/science/article/pii/S2667305325000675.

[86] Nilsson NJ (1986) Probabilistic logic. *Artificial intelligence* 28(1): 71–87. URL https://www.sciencedirect.com/science/article/pii/0004370286900317.

[87] Ozolins E, Freivalds K, Draguns A, Gaile E, Zakovskis R and Kozlovics S (2022) Goal-aware neural sat solver. In: *2022 International joint conference on neural networks (IJCNN)*. IEEE, pp. 1–8. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9892733.

[88] Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, Shamseer L, Tetzlaff JM, Akl EA, Brennan SE, Chou R, Glanville J, Grimshaw

JM, Hróbjartsson A, Lalu MM, Li T, Loder EW, Mayo-Wilson E, McDonald S, McGuinness LA, Stewart LA, Thomas J, Tricco AC, Welch VA, Whiting P and Moher D (2021) The prisma 2020 statement: an updated guideline for reporting systematic reviews. *BMJ* 372. DOI:10.1136/bmj.n71. URL https://www.bmj.com/content/372/bmj.n71.

[89] Pelosi D, Cacciagrano D and Piangerelli M (2025) Explainability and interpretability in concept and data drift: A systematic literature review. *Algorithms* 18(7). DOI:10.3390/a18070443. URL https://www.mdpi.com/1999-4893/18/7/443.

[90] Pinkas G (1991) Propositional non-monotonic reasoning and inconsistency in symmetric neural networks. *In IJCAI* 91: 525–530. URL https://openscholarship.wustl.edu/cgi/viewcontent.cgi?article=1622&context=cse_research.

[91] Pinkas G (1991) Symmetric neural networks and propositional logic satisfiability. *Neural Computation* 3(2): 282–291. URL https://doi.org/10.1162/neco.1991.3.2.282.

[92] Pinkas G (1995) Reasoning, nonmonotonicity and learning in connectionist networks that capture propositional knowledge. *Artificial Intelligence* 77(2): 203–247. DOI:10.1016/0004-3702(94)00032-V.

[93] Plaisted DA (2014) Automated theorem proving. *Wiley Interdisciplinary Reviews: Cognitive Science* 5(2): 115–128. DOI:10.1002/wcs.1269. URL https://doi.org/10.1002/wcs.1269.

[94] Poole DL and Mackworth AK (2010) *Artificial Intelligence: foundations of computational agents*. Cambridge University Press. URL https://www.cambridge.org/highereducation/books/artificial-intelligence/C113F6CE284AB00F5489EBA5A59B93B7#overview.

[95] Rasga J and Sernadas C (2022) Adding abductive reasoning to a propositional logic. *Journal of Logic and Computation* 32(4): 697–728. URL https://academic.oup.com/logcom/article/32/4/697/6457177.

[96] Rasga M, Vukmirović P and Blanchette JC (2019) Efficient verified (un)sat certificate checking. *Journal of Automated Reasoning* 63: 1129–1163. DOI:10.1007/s10817-019-09525-z.

[97] Renkhoff J, Feng K, Meier-Doernberg M, Velasquez A and Song HH (2024) A survey on verification and validation, testing and evaluations of neurosymbolic artificial intelligence. *IEEE Transactions on Artificial Intelligence* 5(8): 3765–3779. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10385139.

[98] Riegel R, Gray A, Luus F, Khan N, Makondo N, Akhalwaya IY, Qian H, Fagin R, Barahona F, Sharma U et al. (2020) Logical neural networks. *arXiv preprint arXiv:2006.13155* URL https://arxiv.org/pdf/2006.13155.

[99] Ritz BM and Hofacker GL (1996) Informational properties of neural nets performing algorithmic and logical tasks. *Biological Cybernetics* 74(6): 549–555. DOI:10.1007/BF00209426. URL https://doi.org/10.1007/BF00209426.

[100] Rocktäschel T and Riedel S (2016) Learning knowledge base inference with neural theorem provers. In: *Proceedings of the 5th workshop on automated knowledge base construction*. pp. 45–50. URL https://aclanthology.org/W16-1309.pdf.

[101] Rocktäschel T and Riedel S (2017) End-to-end differentiable proving. *Advances in neural information processing systems* 30. URL https://proceedings.neurips.cc/paper/2017/file/b2ab001909a8a6f04b51920306046ce5-Paper.pdf.

[102] Rollini SF, Bruttomesso R, Sharygina N and Tsitovich A (2014) Resolution proof transformation for compression and interpolation. *Formal Methods in System Design* 45(1): 1–41. URL https://arxiv.org/pdf/1307.2028.

[103] Russell SJ and Norvig P (2010) *Artificial Intelligence: A Modern Approach*. 3 edition. Upper Saddle River, NJ: Prentice Hall.

[104] Sander J, Yu CEJ, Jalaian B and Bastian ND (2024) Uncertainty-quantified neurosymbolic ai for open set recognition in network intrusion detection. In: *MILCOM 2024 - 2024 IEEE Military Communications Conference (MILCOM)*. pp. 13–18. DOI:10.1109/MILCOM61039.2024.10773953.

[105] Schechter J and Enoch D (2006) Meaning and justification: The case of modus ponens. *Noûs* : 687–715URL https://www.jstor.org/stable/4093984.

[106] Seda AK (2006) On the integration of connectionist and logic-based systems. *Electronic Notes in Theoretical Computer Science* 161: 109–130. DOI:https://doi.org/10.1016/j.entcs.2006.04.028. URL https://www.sciencedirect.com/science/article/pii/S1571066106004038. Proceedings of the Third Irish Conference on the Mathematical Foundations of Computer Science and Information Technology (MFCSIT 2004).

[107] Selman B, Kautz HA, Cohen B et al. (1993) Local search strategies for satisfiability testing. *Cliques, coloring, and satisfiability* 26: 521–532. URL http://eda.ee.ucla.edu/member_only/FPGA/SAT-Mapping/walksat-dimacs93.pdf.

[108] Selman B, Levesque HJ and Mitchell D (1992) A new method for solving hard satisfiability problems. In: *Proceedings of AAAI*. Menlo Park, CA / Cambridge, MA: AAAI Press, pp. 440–446. URL https://dl.acm.org/doi/10.5555/1867135.1867203.

[109] Selsam D and Bjørner N (2019) Guiding high-performance sat solvers with unsat-core predictions. In: *International conference on theory and applications of satisfiability testing*. Springer, pp. 336–353. URL https://arxiv.org/pdf/1903.04671.

[110] Selsam D, Lamm M, Bünz B, Liang P, de Moura L and Dill DL (2019) Learning a sat solver from single-bit supervision. In: *International Conference on Learning Representations*. URL https://openreview.net/forum?id=HJMC_iA5tm.

[111] Shannon CE (1948) A mathematical theory of communication. *Bell System Technical Journal* 27(3): 379–423. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6773024.

[112] Shi S, Chen H, Ma W, Mao J, Zhang M and Zhang Y (2020) Neural logic reasoning. In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. pp. 1365–1374. URL https://dl.acm.org/doi/pdf/10.1145/3340531.3411949.

[113] Silva A and Gombolay M (2021) Encoding human domain knowledge to warm start reinforcement learning. In: *Proceedings of the AAAI conference on artificial intelligence*, volume 35. pp. 5042–5050. URL https://ojs.aaai.org/index.php/AAAI/article/view/16638.

[114] Silver D, Schrittwieser J, Simonyan K, Antonoglou I, Huang A, Guez A, Hubert T, Baker L, Lai M, Bolton A et al. (2017) Mastering the game of go without human knowledge. *nature* 550(7676): 354–359. URL https://www.nature.com/articles/nature24270.pdf.

[115] Smith P (2020) Soundness and completeness. In: *An Introduction to Formal Logic*. Cambridge University Press, pp. 402–411. URL https://www.logicmatters.net/resources/pdfs/IFL2_LM.pdf.

[116] Staffa M, Rossi S, Gregorio MD and Burattini E (2011) Thresholds tuning of a neuro-symbolic net controlling a behavior-based robotic system. In: Verleysen M (ed.) *ESANN 2011 Proceedings: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. Bruges, Belgium: i6doc.com. ISBN 978-2-87419-044-5, pp. 159–164. URL http://www.i6doc.com/en/livre/?GCOI=28001100817300.

[117] Suda M (2021) Improving enigma-style clause selection while learning from history. In: Platzer A and Sutcliffe G (eds.) *Automated Deduction – CADE 28*. Cham: Springer International Publishing. ISBN 978-3-030-79876-5, pp. 543–561. URL https://link.springer.com/chapter/10.1007/978-3-030-79876-5_31.

[118] Suda M (2025) Efficient neural clause-selection reinforcement. URL https://arxiv.org/abs/2503.07792.

[119] Sun L, Gerault D, Benamira A and Peyrin T (2020) Neurogift: Using a machine learning based sat solver for cryptanalysis. In: *Cyber Security Cryptography and Machine Learning*. Springer, pp. 62–84.

[120] Towell GG and Shavlik JW (1994) Knowledge-based artificial neural networks. *Artificial Intelligence* 70(1): 119–165. DOI:https://doi.org/10.1016/0004-3702(94)90105-8. URL https://www.sciencedirect.com/science/article/pii/0004370294901058.

[121] Towell GG and Shavlik JW (1994) Knowledge-based artificial neural networks. *Artificial Intelligence* 70(1–2): 119–165. DOI:10.1016/0004-3702(94)90105-8.

[122] Towell GG, Shavlik JW and Noordewier MO (1990) Refinement of approximate domain theories by knowledge-based neural networks : 861–866URL https://dl.acm.org/doi/abs/10.5555/1865609.1865629.

[123] Townsend J, Chaton T and Monteiro JM (2019) Extracting relational explanations from deep neural networks: A survey from a neural-symbolic perspective. *IEEE transactions on neural networks and learning systems* 31(9): 3456–3470. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8889997.

[124] Tran S, Mota E and Garcez Ad (2025) Reasoning in neurosymbolic ai. *arXiv preprint arXiv:2505.20313* URL https://arxiv.org/pdf/2505.20313?

[125] Tran SN (2017) Propositional knowledge representation and reasoning in restricted boltzmann machines. *arXiv preprint arXiv:1705.10899* URL https://arxiv.org/pdf/1705.10899.

[126] Tran SN (2017) Unsupervised neural-symbolic integration. *arXiv preprint arXiv:1706.01991* URL https://arxiv.org/pdf/1706.01991.

[127] Tran SN and d'Avila Garcez A (2021) Logical boltzmann machines. URL https://arxiv.org/pdf/2112.05841.

[128] Tran SN and Garcez Ad (2023) Neurosymbolic reasoning and learning with restricted boltzmann machines 37(5): 6558–6565. URL https://ojs.aaai.org/index.php/AAAI/article/view/25806.

[129] Trevisan L (2001) Notes on the np-completeness of circuit-sat. URL https://theory.stanford.edu/~trevisan/cs154-12/notenp.pdf.

[130] Tsakonas A, Aggelis V, Karkazis I and Dounias G (2004) An evolutionary system for neural logic networks using genetic programming and indirect encoding. *Journal of Applied Logic* 2(3): 349–379. DOI:https://doi.org/10.1016/j.jal.2004.03.005. URL https://www.sciencedirect.com/science/article/pii/S1570868304000199.

[131] Unno H, Torii S and Sakamoto H (2017) Automating induction for solving horn clauses. In: *International Conference on Computer Aided Verification (CAV), Lecture Notes in Computer Science*, volume 10427. Springer, pp. 571–591. DOI:10.1007/978-3-319-63390-9_30.

[132] Vaezipoor P, Lederman G, Wu Y, Grosse R and Bacchus F (2020) Learning clause deletion heuristics with reinforcement learning. In: *5th Conference on Artificial Intelligence and Theorem*

Proving. p. 80. URL https://aitp-conference.org/2020/abstract/paper_25.pdf.

[133] Wan Z, Liu CK, Yang H, Li C, You H, Fu Y, Wan C, Krishna T, Lin Y and Raychowdhury A (2024) Towards cognitive ai systems: a survey and prospective on neuro-symbolic ai. *arXiv preprint arXiv:2401.01040* URL https://arxiv.org/pdf/2401.01040.

[134] Wang PW, Donti P, Wilder B and Kolter Z (2019) Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In: *International Conference on Machine Learning*. PMLR, pp. 6545–6554. URL https://proceedings.mlr.press/v97/wang19e/wang19e.pdf.

[135] Wang W, Hu Y, Tiwari M, Khurshid S, McMillan K and Miikkulainen R (2024) Neuroback: Improving cdcl sat solving using graph neural networks. In: *The Twelfth International Conference on Learning Representations*. URL https://openreview.net/forum?id=samyfu6G93.

[136] Wang W, Yang Y and Wu F (2024) Towards data-and knowledge-driven ai: a survey on neuro-symbolic computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10721277.

[137] Weitze MD and Hofacker GL (1995) The implementation of propositional logic in random neural networks. In: *Proceedings of the 7th International Conference on Tools with Artificial Intelligence*. Springer, pp. 54–61. URL https://link.springer.com/chapter/10.1007/3-540-59497-3_173.

[138] Wendelken C and Shastri L (2004) Multiple instantiation and rule mediation in shruti. *Connection Science* 16(3): 211–217. URL https://www.tandfonline.com/doi/pdf/10.1080/09540090412331311932.

[139] Wetzler N, Heule MJH and Hunt WA (2014) Drat-trim: Efficient checking and trimming using expressive clausal proofs. In: Sinz C and Egly U (eds.) *Theory and Applications of Satisfiability Testing – SAT 2014*. Springer International Publishing. ISBN 978-3-319-09284-3, pp. 422–429. URL https://link.springer.com/chapter/10.1007/978-3-319-09284-3_31.

[140] Xu J, Zhang Z, Friedman T, Liang Y and Van den Broeck G (2018) A semantic loss function for deep learning with symbolic knowledge. In: *International Conference on Machine Learning*. PMLR, pp. 5502–5511. URL https://proceedings.mlr.press/v80/xu18h/xu18h.pdf.

[141] Yolcu E and Póczos B (2019) Learning local search heuristics for boolean satisfiability. *Advances in Neural Information Processing Systems* 32. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/12e59a33dea1bf0630f46edfe13d6ea2-Paper.pdf.

[142] Yu D, Yang B, Liu D, Wang H and Pan S (2023) A survey on neural-symbolic learning systems. *Neural Networks* 166: 105–126. DOI: https://doi.org/10.1016/j.neunet.2023.06.028. URL https://www.sciencedirect.com/science/article/pii/S0893608023003398.

[143] Zhang H, Chen W et al. (2025) Hypersat: Unsupervised hypergraph neural networks for weighted maxsat problems. *arXiv preprint arXiv:2504.11885* URL https://arxiv.org/pdf/2504.11885.

[144] Zhang H, Li LH, Meng T, Chang KW and Broeck GVd (2022) On the paradox of learning to reason from data. URL https://arxiv.org/pdf/2205.11502.

[145] Zhang H and Zhang J (2022) *Logic in Computer Science*. PhD Thesis, Graduate College, The University of Iowa, Iowa City, Iowa, USA. URL https://homepage.divms.uiowa.edu/~hzhang/c188/notes/book.pdf.

[146] Zhang Y, Zhen HL, Yuan M and Yu B (2024) Diffsat: Differential maxsat layer for sat solving. In: *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*. pp. 1–7. URL https://dl.acm.org/doi/pdf/10.1145/3676536.3676748.