Modeling Patterns for Neural-Symbolic Reasoning Using Energy-based Models

Charles Dickens[1], **Connor Pryor**[1], **Lise Getoor**[1]

[1] University of California Santa Cruz
cadicken@ucsc.edu, cfpryor@ucsc.edu, getoor@ucsc.edu

Abstract

Neural-symbolic (NeSy) AI strives to empower machine learning and large language models with fast, reliable predictions that exhibit commonsense and trustworthy reasoning by seamlessly integrating neural and symbolic methods. With such a broad scope, several taxonomies have been proposed to categorize this integration, emphasizing knowledge representation, reasoning algorithms, and applications. We introduce a knowledge representation-agnostic taxonomy focusing on the neural-symbolic interface capturing methods that reason with probability, logic, and arithmetic constraints. Moreover, we derive expressions for gradients of a prominent class of learning losses and a formalization of reasoning and learning. Through a rigorous empirical analysis spanning three tasks, we show NeSy approaches reach up to a 37% improvement over neural baselines in a semi-supervised setting and a 19% improvement over GPT-4 on question-answering.

1 Introduction

Modern AI methods often employ black-box models, such as deep neural networks, that excel at perception and generation with low-level data (e.g., pixels or tokens) but lack consistent and interpretable reasoning capabilities. This significantly hinders the utility and trustworthiness of these models. On the other hand, symbolic frameworks for performing logical or mathematical reasoning exist with complementary properties; they are reliable and consistent but struggle with low-level perception and generation. The field of neural-symbolic (NeSy) AI (d'Avila Garcez, Broda, and Gabbay 2002; d'Avila Garcez, Lamb, and Gabbay 2009) has emerged with the goal of developing systems that realize a synergy between neural and symbolic methods. As a result, the field of NeSy AI is experiencing rapid growth, with new models regularly being introduced (Manhaeve et al. 2021; Xu et al. 2018; Badreddine et al. 2022; Ahmed et al. 2022; Pryor et al. 2023). A number of taxonomies have been proposed to categorize these systems (Bader and Hitzler 2005; d'Avila Garcez et al. 2019; De Raedt et al. 2020; Lamb et al. 2020; Besold et al. 2022; Giunchiglia, Stoian, and Lukasiewicz 2022). Among these efforts, the emphasis has traditionally been on knowledge representation, reasoning algorithms, and applications.

This work introduces a new NeSy taxonomy that is knowledge representation-agnostic, capturing methods that reason with probability, logic, and arithmetic constraints. We introduce a mathematical programming abstraction of symbolic reasoning based on the widely used energy-based modeling (EBM) paradigm (LeCun et al. 2006). Specifically, we build upon NeSy-EBMs: a family of EBMs defined by energy functions that are compositions of neural and symbolic components (Pryor et al. 2023).

We organize approaches for connecting the neural and symbolic components of a NeSy system into three categories: *deep symbolic variables*, *deep symbolic parameters*, and *deep symbolic potentials*. These categories are differentiated by the role of the neural component in formulating the mathematical program for reasoning and vary with increasing expressivity and complexity. We present an empirical analysis of the strengths and use cases of each of the three modeling patterns, including semi-supervised learning, graph node labeling with a mixture of experts, and question answering with a large language model (LLM).

Additionally, we derive expressions for gradients of a general class of NeSy learning losses. The gradient expressions motivate design decisions for the neural-symbolic interface to ensure continuity properties needed for principled gradient-based learning. We use the gradients to perform end-to-end learning in our empirical analysis for NeSy models making categorical predictions.

Our contributions include: 1) The introduction of a novel taxonomy for NeSy modeling patterns and a general formalization of reasoning and learning. 2) Expressions for gradients of a prominent class of NeSy learning losses and a set of conditions to ensure principled gradient-based learning. 3) An extensive empirical analysis of the NeSy modeling patterns reaching up to 37% performance improvements over neural baselines in a semi-supervised problem and 19% improvement of GPT-4 on a question-answering dataset.

2 Related Work

NeSy AI is an active area of research that empowers neural models with domain knowledge and reasoning via the integration with symbolic systems (d'Avila Garcez, Broda, and Gabbay 2002; d'Avila Garcez, Lamb, and Gabbay 2009). This line of work has gained significant attention, resulting in a surge of novel NeSy AI models, including, but not

limited to, the works of Yang, Yang, and Cohen (2017), Evans and Grefenstette (2018), Xu et al. (2018), (Yang, Ishay, and Lee 2020), Manhaeve et al. (2021), Badreddine et al. (2022), Ahmed et al. (2022), Cornelio et al. (2023), and Pryor et al. (2023). Bader and Hitzler (2005), d'Avila Garcez et al. (2019), and most recently Besold et al. (2022) provide extensive surveys using dimensions including knowledge representation, neural-symbolic integration, and application to compare and describe methods. Similarly, the works of De Raedt et al. (2020) and Lamb et al. (2020) propose taxonomies to connect NeSy to statistical relational learning and graph neural networks, respectively. Focused taxonomies and investigations are put forward by Giunchiglia, Stoian, and Lukasiewicz (2022) and van Krieken, Acar, and van Harmelen (2022) for deep learning with constraints and symbolic knowledge representations and Dash et al. (2022) for integrating domain knowledge into deep neural networks. Finally, Marconato et al. (2023) characterizes common reasoning mistakes made by NeSy models.

3 Neural-Symbolic Energy-Based Models

Our modeling taxonomy and learning results build on the neural-symbolic energy-based model (NeSy-EBM) framework introduced by Pryor et al. (2023). NeSy-EBMs are a family of EBMs (LeCun et al. 2006) that integrate deep architectures with explicit encodings of symbolic relations via an energy function. The energy function defines a mathematical program that is minimized to obtain predictions that are consistent with domain knowledge and commonsense.

As diagrammed in Fig. 1, a NeSy-EBM energy function is a composition of a neural and symbolic component, represented by the functions $\mathbf{g}_{nn}$ and $\mathbf{g}_{sy}$, respectively. The neural component is a deep model (or collection of deep models) parameterized by weights from a domain $\mathcal{W}_{nn}$, that takes a neural input from a domain $\mathcal{X}_{nn}$ and outputs a real-valued vector of dimension d_{nn}. The symbolic component is a function encoding domain knowledge and parameterized by weights from a domain $\mathcal{W}_{sy}$ that maps inputs from a domain $\mathcal{X}_{sy}$, target (or output) variables from $\mathcal{Y}$, and neural outputs to a scalar value. Intuitively, the symbolic component measures the compatibility of targets, inputs, and neural outputs with domain knowledge. Formally,

Definition 3.1. *A **NeSy-EBM energy function** is a mapping parameterized by neural and symbolic weights from domains $\mathcal{W}_{nn}$ and $\mathcal{W}_{sy}$, respectively, and quantifies the compatibility of a target variable from a domain $\mathcal{Y}$ and neural and symbolic inputs from the domains $\mathcal{X}_{nn}$ and $\mathcal{X}_{sy}$, respectively, with a single scalar value:*

$$E : \mathcal{Y} \times \mathcal{X}_{sy} \times \mathcal{X}_{nn} \times \mathcal{W}_{sy} \times \mathcal{W}_{nn} \to \mathbb{R}. \tag{1}$$

*A NeSy-EBM energy function is a composition of a **neural** and **symbolic component**. The neural component is parameterized by neural weights and outputs a real-valued vector of dimension d_{nn}: $\mathbf{g}_{nn} : \mathcal{X}_{nn} \times \mathcal{W}_{nn} \to \mathbb{R}^{d_{nn}}$. The symbolic component maps the symbolic variables, symbolic parameters, and a real-valued vector of dimension d_{nn} to a scalar value: $g_{sy} : \mathcal{Y} \times \mathcal{X}_{sy} \times \mathcal{W}_{sy} \times \mathbb{R}^{d_{nn}} \to \mathbb{R}$. Then, the NeSy-EBM energy function is:*

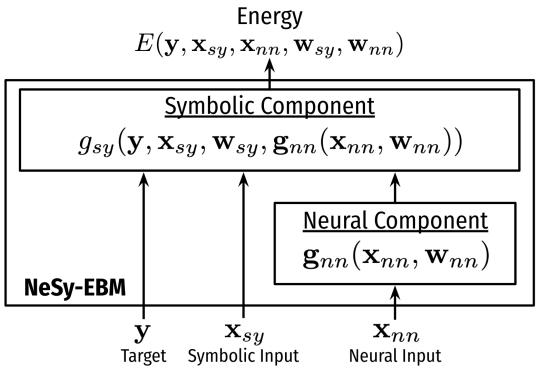

Figure 1: A neural-symbolic energy-based model.

$$\begin{aligned} &E(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{sy}, \mathbf{w}_{nn}) \\ &\quad := g_{sy}(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})). \end{aligned} \tag{2}$$

Given inputs and parameters $(\mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{sy}, \mathbf{w}_{nn}) \in \mathcal{X}_{sy} \times \mathcal{X}_{nn} \times \mathcal{W}_{sy} \times \mathcal{W}_{nn}$, NeSy-EBM energy functions can be used to define several inference tasks. In this work, we focus on finding targets that minimize the energy function, i.e., prediction or decision-making.

$$\arg\min_{\hat{\mathbf{y}} \in \mathcal{Y}} E(\hat{\mathbf{y}}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{sy}, \mathbf{w}_{nn}). \tag{3}$$

NeSy-EBM prediction captures a broad range of reasoning frameworks, including probabilistic, logical, and arithmetic. For instance, it can represent complex NeSy systems such as DeepProbLog (Manhaeve et al. 2021), LTNs (Badreddine et al. 2022), and NeuPSL (Pryor et al. 2023), to name a few.

4 NeSy-EBM Modeling Patterns

This section introduces a taxonomy of NeSy modeling patterns determined by the neural-symbolic interface, i.e., how the neural outputs are utilized in the symbolic component to define the prediction program in (3). To formalize the modeling patterns, we introduce a layer of abstraction we refer to as *symbolic potentials*, denoted by ψ, and *symbolic potential sets*, denoted by $\mathbf{\Psi}$. Symbolic potentials organize the arguments of the symbolic component by the role they play in formulating the prediction program and belong to a symbolic potential set. The specification of the set of symbolic potentials and the domains of potentials belonging to the set defines a modeling pattern. Formally, a symbolic potential, ψ, is a function random variables from a domain RV_ψ and parameters from a domain $Params_\psi$, and outputs a scalar value: $\psi : RV_\psi \times Params_\psi \to \mathbb{R}$. Further, $\mathbf{\Psi}$ denotes a set of potential functions that is indexed by the set $\mathbf{J}_{\mathbf{\Psi}}$.

Using the abstraction of symbolic potentials to formalize the neural-symbolic interface, we introduce three modeling patterns in the following subsections that vary with increasing sophistication: *deep symbolic variables (DSVar)*, *deep symbolic parameters (DSPar)*, and *deep symbolic potentials (DSPot)*. Throughout, we connect the modeling patterns to prominent NeSy systems and provide illustrative examples describing applications studied in our empirical analysis.

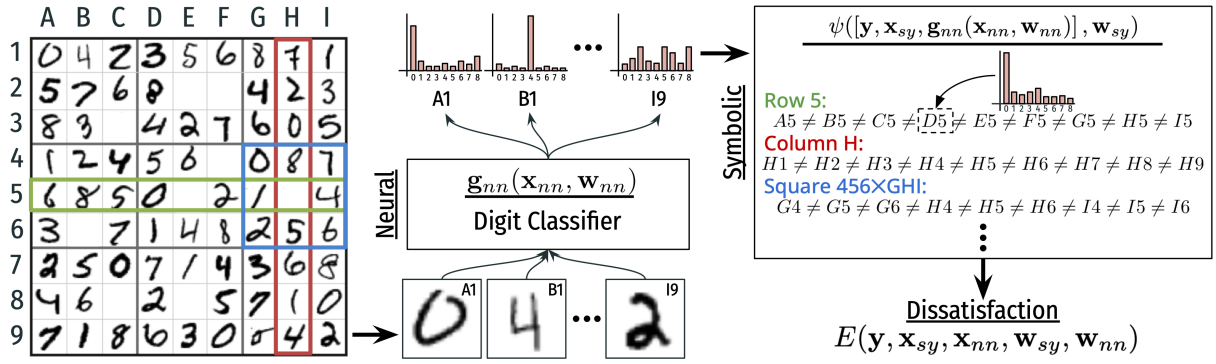

Figure 2: A deep symbolic variables model for solving a Sudoku board constructed from handwritten digits. The neural component classifies handwritten digits. Then the symbolic component uses the digit classifications and the rules of Sudoku to fill in the empty cells. See Ex. 4.1 extended details and Section 6.1 for an empirical analysis.

4.1 Deep Symbolic Variables

The deep symbolic variables pattern is an efficient approach to training a neural component with a loss that captures domain knowledge. Concisely, the neural component directly predicts the values of targets in a single symbolic potential. In other words, there is a one-to-one mapping from the neural output to the targets. This approach typically yields the simplest prediction program as the neural model fixes a subset of the decision variables. However, for the same reason, the symbolic component cannot be used to resolve constraint violations made by the neural component during prediction. Rather, this modeling pattern relies on learning to train a neural component that consistently adheres to constraints. Representative methods following this paradigm include semantic loss networks (Xu et al. 2018) and learning with logical constraints (Giunchiglia, Stoian, and Lukasiewicz 2022).

Formally, a symbolic potential function is a singleton $\mathbf{\Psi} = \{\psi\}$ with a trivial index set $\mathbf{J}_{\mathbf{\Psi}} = \{1\}$ such that $\mathbf{\Psi}_1 = \psi$. Further, the neural prediction is treated as a random variable by the symbolic potential; thus $RV_\psi = \mathcal{Y} \times \mathcal{X}_{sy} \times \mathbb{R}^{d_{nn}}$, and the domain of the symbolic parameters is the symbolic weights, $Params_\psi = \mathcal{W}_{sy}$. The neural component controls the NeSy-EBM prediction via an indicator function:

$$\chi_{\mathcal{Y}}(\mathbf{y}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})) := \begin{cases} 0 & \mathbf{y}_i = \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})_i, \forall i \in \{1, \cdots, d_{nn}\} \\ \infty & \text{o.w.} \end{cases}, \tag{4}$$

where $\mathbf{y}_i$ and $\mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})_i$ denote the i'th entry of the random variable and neural output vectors, respectively. The indicator function $\chi_{\mathcal{Y}}$ is added to the symbolic potential to define the symbolic component. In this way, infinite energy is assigned to random variable values that do not match the neural model's predictions and will not be predicted by the NeSy-EBM. Therefore, the symbolic component expressed via the symbolic potential is:

$$\begin{aligned} &g_{sy}(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})) \\ &:= \psi([\mathbf{y}, \mathbf{x}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})], \mathbf{w}_{sy}) + \chi_{\mathcal{Y}}(\mathbf{y}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})), \end{aligned} \tag{5}$$

where $[\cdot]$ denotes concatenation. This pattern is demonstrated in the following example.

Example 4.1. *Visual Sudoku puzzle solving is the problem of recognizing handwritten digits in non-empty puzzle cells and reasoning with the rules of Sudoku (no repeated digits in any row, column, or box) to fill in empty cells. Fig. 2 shows a partially complete Sudoku puzzle created with MNIST images (LeCun et al. 1998) and a NeSy-EBM designed for visual Sudoku solving. The neural component is a digit classifier predicting the label of MNIST images, and the symbolic component quantifies rule violations.*

Formally, the target variables, $\mathbf{y}$, are the categorical labels of both the handwritten digits and the puzzle's empty entries. The symbolic inputs, $\mathbf{x}_{sy}$, indicate whether two puzzle positions are in the same row, column, or box. The neural model, $\mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})$, is the categorical labels of the handwritten digits predicted by the neural component. Then, the symbolic parameters, $\mathbf{w}_{sy}$, are used to shape the single symbolic potential function, ψ, that quantifies the total amount of Sudoku rule violations.

The deep symbolic variables modeling pattern is applied in a semi-supervised learning setting to fit neural parameters with a partially labeled dataset in Section 6.1. However, as neural model predictions determine a subset of the target values, the model cannot resolve rule violations. When the neural model predicts digit labels violating a Sudoku rule, the predicted target variables also violate the rule.

4.2 Deep Symbolic Parameters

The deep symbolic parameters modeling pattern allows targets and neural predictions to be unequal or represent different concepts. Succinctly, the neural component is applied as a parameter in the symbolic potential. This pattern allows the symbolic component to correct constraint violations made by the neural component during prediction. Prominent NeSy frameworks supporting this technique include DeepProbLog (Manhaeve et al. 2021), semantic probabilistic layers (Ahmed et al. 2022), logic tensor networks (Badreddine et al. 2022), and neural probabilistic soft logic (Pryor et al. 2023).

Formally, there is only one potential in the potential function set denoted by $\mathbf{\Psi} = \{\psi\}$. Moreover, the index set of the potential function set is simply $\mathbf{J}_{\mathbf{\Psi}} = \{1\}$ such that

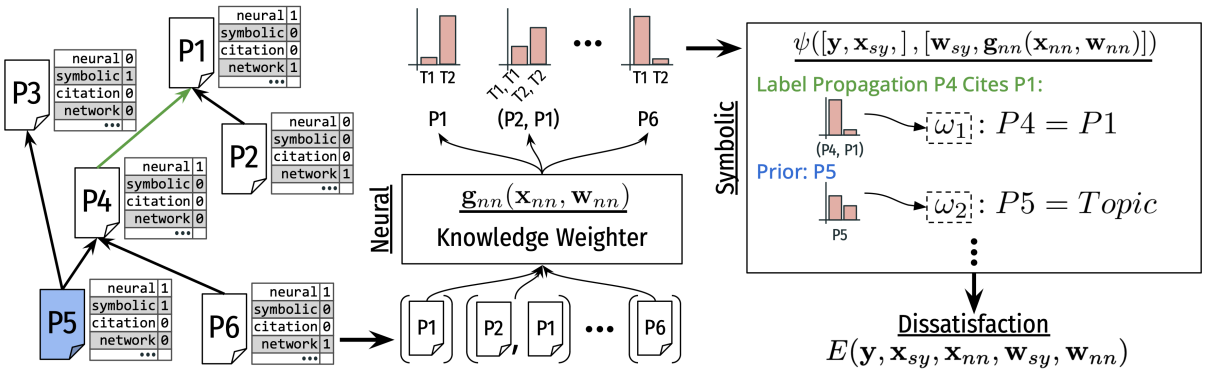

Figure 3: A deep symbolic parameters model for citation network node classification. The symbolic component is a mixture of experts model that combines weighted arithmetic constraints. The neural component uses paper content to weigh the importance of satisfying an arithmetic constraint. See Ex. 4.2 extended details and Section 6.2 for an empirical analysis.

$\mathbf{\Psi}_1 = \psi$. Further, the symbolic potential parameters are the symbolic weights and the neural predictions, i.e., the parameter domain is $Params_\psi = \mathcal{W}_{sy} \times \mathbb{R}^{d_{nn}}$. The symbolic random variables are the targets and the symbolic inputs, i.e., $RV_\psi = \mathcal{Y} \times \mathcal{X}_{sy}$. Thus, the symbolic component expressed via the single symbolic potential is:

$$g_{sy}(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})) \\ := \psi([\mathbf{y}, \mathbf{x}_{sy},], [\mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})]). \tag{6}$$

This pattern is demonstrated in the following example.

Example 4.2. *Citation network node classification is the task of predicting the topic of papers in a network where nodes are papers and edges are citations. A model may use the network structure, content, and topic labels in a paper's neighborhood for prediction. Fig. 3 shows a citation network and a NeSy-EBM designed for node classification. The symbolic component is a collection of weighted arithmetic constraints. One constraint represents the heuristic that two papers connected by a citation in the network have the same topic, and another uses a neural network to predict a topic for each paper using only its content. If a constraint is violated by a random variable, then a cost proportional to the weight of the violated constraint will be added to the energy function. The neural component predicts the weights of each constraint given the paper content. In other words, the NeSy-EBM is a mixture of experts. Each arithmetic constraint in the symbolic component is an expert weighted by the neural component, and experts are combined to formulate a mathematical program to produce a single output.*

Formally, the targets, $\mathbf{y}$, are the paper topics, and the symbolic inputs, $\mathbf{x}_{sy}$, are citation links. The neural model, $\mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})$, predicts the weights of every constraint. Finally, the symbolic parameters, $\mathbf{w}_{sy}$, are used to shape the single symbolic potential function, ψ, quantifying the weighted dissatisfaction of the constraints.

In Section 6.2, the deep symbolic parameters modeling pattern is applied to a synthetic citation network setting as described in the example above. With the deep symbolic variables and parameters modeling patterns, the NeSy-EBM has a single fixed symbolic potential. This is a powerful technique for well-defined and dedicated tasks; however, it is less applicable to open-ended settings where the relevant domain knowledge is dependent on context. To address this challenge, the next modeling pattern leverages a generative neural component to sample a symbolic potential.

4.3 Deep Symbolic Potentials

Deep-symbolic potentials, the most advanced pattern, empower deep models and LLMs with symbolic reasoning tools. Input data is used as context to retrieve relevant domain knowledge and formulate a program to perform inference in open-ended problems. Specifically, the neural component is a generative model that samples symbolic potentials from a set. This modeling pattern is best represented by the Logic-LM pipeline proposed by Pan et al. (2023). In Logic-LM, an LLM is provided a problem and goal in the form of a prompt and generates syntax for a symbolic reasoning program that is executed and interpreted to obtain an answer. Any NeSy framework with syntax and semantics for instantiating symbolic potentials can be integrated into a Logic-LM style pipeline to achieve this pattern.

Formally, let the potential function set, $\mathbf{\Psi}$, be the set of potential functions that can be created by a NeSy framework. $\mathbf{\Psi}$ is indexed by the output of the neural component, i.e., $\mathbf{J}_\Psi = Range(\mathbf{g}_{nn})$ and $\mathbf{\Psi}_{\mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})}$ is the potential function indexed by the neural prediction. The random variable and parameter domains of the sampled symbolic potential are $RV_\psi = \mathcal{Y} \times \mathcal{X}_{sy}$, and $Params_\psi = \mathcal{W}_{sy}$, respectively. Then, the symbolic component expressed via the symbolic potential is:

$$g_{sy}(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})) \\ := \mathbf{\Psi}_{\mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})}([\mathbf{y}, \mathbf{x}_{sy}], \mathbf{w}_{sy}), \tag{7}$$

This pattern is demonstrated in the following example.

Example 4.3. *Question answering is the problem of giving a response to a question posed in natural language. Fig. 4 shows a set of word problems asking for the order of a set of objects given information expressed in natural language and a NeSy-EBM designed for question answering. The neural component is an LLM that is provided a word problem*

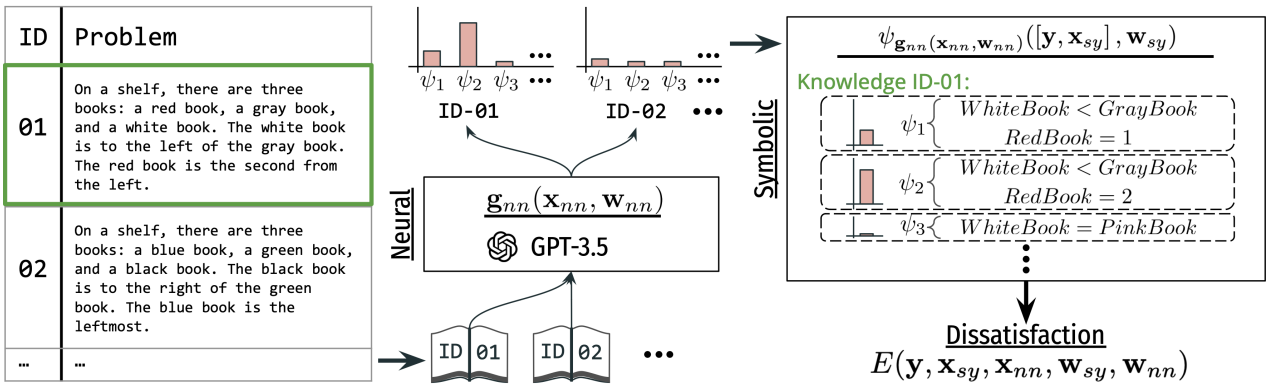

Figure 4: A deep symbolic potential model for answering questions about a set of objects' order described in natural language. The neural component is a generative model that samples a symbolic potential. The symbolic potential performs deductive reasoning to answer the question. See Ex. 4.3 extended details and Section 6.3 for an empirical analysis.

and generates syntax for a NeSy framework to instantiate a symbolic component used to perform deductive reasoning.

Formally, the target variables, $\mathbf{y}$*, represent object positions, and there is no symbolic input,* $\mathbf{x}_{sy}$*, in this example. The neural input,* $\mathbf{x}_{nn}$*, is a prompt that includes the natural language word problem. The neural model,* $\mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})$*, is an LLM that generates syntax for a declarative symbolic modeling framework that creates the symbolic potential. For instance, the symbolic potential generated by the neural model,* $\boldsymbol{\Psi}_{\mathbf{g}_{nn}(\mathbf{x}_{nn},\mathbf{w}_{nn})}([\mathbf{y}, \mathbf{x}_{sy}], \mathbf{w}_{sy})$*, may be the total amount of violation of arithmetic constraints representing ordering. Finally, the symbolic parameters* $\mathbf{w}_{sy}$ *are used to shape the symbolic potential function.*

In Section 6.3, this modeling pattern is applied to the logical deduction setting as described above.

5 NeSy-EBM Learning

NeSy-EBM learning is finding weights of an energy function, defined in (1), that associates lower energies to targets and neural outputs near their true labels. This section provides expressions for learning gradients in terms of the neural and symbolic components for a general class of loss.

The training dataset, denoted by $\mathcal{S}$, comprises P samples. Each sample $\mathcal{S}_i$, where $i \in \{1, \cdots, P\}$, is a tuple of *inputs*, *labels*, and *latent variable domains*. Sample *inputs* consist of neural inputs, $\mathbf{x}^i_{nn}$ from $\mathcal{X}_{nn}$, and symbolic inputs, $\mathbf{x}^i_{sy}$ from $\mathcal{X}_{sy}$. Similarly, sample *labels* consist of neural and symbolic labels, which are truth values corresponding to a subset of the neural predictions and target variables, respectively. Neural labels, denoted by $\mathbf{t}^i_{nn}$, are $d^i_{nn} \leq d_{nn}$ dimensional real vectors from a domain $\mathcal{T}^i_{nn}$, i.e., $\mathbf{t}^i_{nn} \in \mathcal{T}^i_{nn} \subseteq \mathbb{R}^{d^i_{nn}}$. Target labels, denoted by $\mathbf{t}^i_{\mathcal{Y}}$, are from a domain $\mathcal{T}^i_{\mathcal{Y}}$ that is a subspace of the target domain $\mathcal{Y}$, i.e., $\mathbf{t}^i_{\mathcal{Y}} \in \mathcal{T}^i_{\mathcal{Y}}$. Lastly, the neural and symbolic *latent variable domains* are subspaces of the range of the neural component and the target domain, respectively, corresponding to the set of unlabeled variables. The neural range and target domain are supersets of the Cartesian products of their respective latent variable and label domains, i.e., $\mathbb{R}^{d^{nn}} \supseteq \mathcal{T}^i_{nn} \times \mathcal{Z}^i_{nn}$ and $\mathcal{Y} \supseteq \mathcal{T}^i_{\mathcal{Y}} \times \mathcal{Z}^i_{\mathcal{Y}}$. The training data is expressed as follows:

$$\mathcal{S} := \{(\mathbf{t}^1_{\mathcal{Y}}, \mathbf{t}^1_{nn}, \mathcal{Z}^1_{nn}, \mathcal{Z}^1_{\mathcal{Y}}, \mathbf{x}^1_{sy}, \mathbf{x}^1_{nn}), \cdots, (\mathbf{t}^P_{\mathcal{Y}}, \mathbf{t}^P_{nn}, \mathcal{Z}^P_{nn}, \mathcal{Z}^P_{\mathcal{Y}}, \mathbf{x}^P_{sy}, \mathbf{x}^P_{nn})\}. \tag{8}$$

A *learning objective*, denoted by $\mathcal{L}$, is a functional that maps an energy function and a training dataset to a scalar value. Formally, let $\mathcal{E}$ be a family of energy functions indexed by weights from $\mathcal{W}_{sy} \times \mathcal{W}_{nn}$:

$$\mathcal{E} := \{E(\cdot, \cdot, \cdot, \mathbf{w}_{sy}, \mathbf{w}_{nn}) \,|\, (\mathbf{w}_{sy}, \mathbf{w}_{nn}) \in \mathcal{W}_{sy} \times \mathcal{W}_{nn}\}. \tag{9}$$

Then, a learning objective is a function $\mathcal{L} : \mathcal{E} \times \{\mathcal{S}\} \to \mathbb{R}$. Further, learning objectives are separable over elements of $\mathcal{S}$ as a sum of *per-sample loss functionals* denoted by L^i for each $i \in \{1, \cdots, P\}$. A loss functional for the sample $\mathcal{S}_i \in \mathcal{S}$ is a function $L^i : \mathcal{E} \times \{\mathcal{S}_i\} \to \mathbb{R}$. With this notation, NeSy-EBM learning is the following minimization problem:

$$\begin{aligned} &\underset{(\hat{\mathbf{w}}_{sy}, \hat{\mathbf{w}}_{nn}) \in \mathcal{W}_{sy} \times \mathcal{W}_{nn}}{\arg\min} \mathcal{L}(E(\cdot, \cdot, \cdot, \mathbf{w}_{sy}, \mathbf{w}_{nn}), \mathcal{S}) \\ &= \underset{(\hat{\mathbf{w}}_{sy}, \hat{\mathbf{w}}_{nn}) \in \mathcal{W}_{sy} \times \mathcal{W}_{nn}}{\arg\min} \sum_{i=1}^{P} L^i(E(\cdot, \cdot, \cdot, \mathbf{w}_{sy}, \mathbf{w}_{nn}), \mathcal{S}_i). \end{aligned} \tag{10}$$

Typically, the energy function parameters are also regularized by a function $\mathcal{R} : \mathcal{W}_{sy} \times \mathcal{W}_{nn} \to \mathbb{R}$, which we omit in the formulation to simplify the exposition.

For every training sample $\mathcal{S}_i \in \mathcal{S}$, the general family of loss functionals we consider are defined using *optimal value-functions*. Generally, an optimal value-function is the minimizing value of an objective defined with the energy over a subset of targets. For instance, the latent optimal value-function, denoted by V, is defined as:

$$\begin{aligned} &V(\mathbf{t}^i_{\mathcal{Y}}, \mathbf{x}^i_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})) \\ &:= \min_{\hat{\mathbf{z}} \in \mathcal{Z}^i_{\mathcal{Y}}} g_{sy}((\mathbf{t}^i_{\mathcal{Y}}, \hat{\mathbf{z}}), \mathbf{x}^i_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})) \end{aligned} \tag{11}$$

An illustration of an example latent optimal value-function is provided in Fig. 5. The latent optimal value-function is the greatest lower bound of the set of symbolic components defined for each latent variable.

In this work, we organize loss functionals into a sum of two losses: *neural* and *value-based*. A neural loss is any

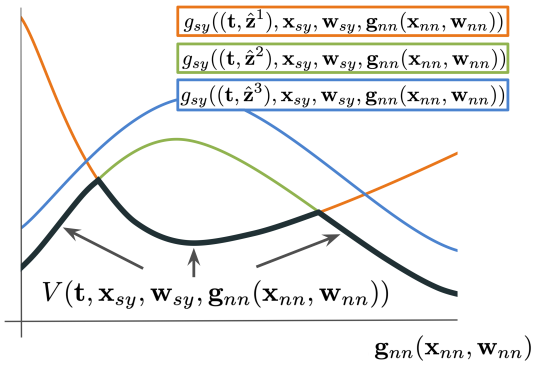

Figure 5: An illustrated example of a latent optimal value-function with a scalar neural component output and a discrete latent variable domain $\mathcal{Z}_{\mathcal{Y}} := \{\hat{\mathbf{z}}^1, \hat{\mathbf{z}}^2, \hat{\mathbf{z}}^3\}$.

differentiable loss (e.g., cross-entropy) that is a function of the neural output and neural labels and is expressed as $L_{NN} : \mathbb{R}^{d_{nn}} \times \mathbb{R}^{d^i_{nn}} \to \mathbb{R}$. A valued-based loss depends on the model weights strictly via optimal value-functions, including but not limited to (11), and are expressed as $L_{Val} : \mathcal{E} \times \{\mathcal{S}_i\} \to \mathbb{R}$. For instance, in our empirical evaluation, we apply the standard *energy loss*:

$$L_{Val}(E(\cdot,\cdot,\cdot,\mathbf{w}_{sy},\mathbf{w}_{nn}),\mathcal{S}^i) \quad (12)$$
$$:= V(\mathbf{t}^i_{\mathcal{Y}}, \mathbf{x}^i_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})).$$

Intuitively, the energy loss is applied to find parameters that result in low energy values for the target labels. Altogether, a loss for a sample $\mathcal{S}_i \in \mathcal{S}$ is:

$$L^i(E(\cdot,\cdot,\cdot,\mathbf{w}_{sy},\mathbf{w}_{nn}),\mathbf{S}_i) \quad (13)$$
$$= L_{NN}(\mathbf{g}_{nn}(\mathbf{x}^i_{nn},\mathbf{w}_{nn}),\mathbf{t}^i_{nn}) + L_{Val}(E(\cdot,\cdot,\cdot,\mathbf{w}_{sy},\mathbf{w}_{nn}),\mathbf{S}_i)$$

Weights that minimize the learning objective are found via gradient-descent or similar algorithms. The gradient of a neural loss with respect to the neural weights is obtained via a standard application of the backpropagation algorithm.

The gradient of a value-based loss with respect to neural and symbolic weights are non-trivial due to the fact that both the energy function and the point the energy function is evaluated at are dependent on the neural output and symbolic weights, as exemplified by the energy loss and the latent-optimal-value-function in (12) and (11), respectively. Nonetheless, Milgrom and Segal (2002) provides a general theorem defining the gradient of optimal value-functions with respect to problem parameters, if they exist. We specialize their result in the following theorem for the latent optimal value-function of NeSy-EBMs.

Theorem 5.1 (Milgrom and Segal (2002) Theorem 1 for NeSy-EBMs)**.** *Consider $\mathbf{w}_{sy} \in \mathcal{W}_{sy}$ and $\mathbf{w}_{nn} \in \mathcal{W}_{nn}$ and the sample $\mathcal{S}_i = (\mathbf{t}^i_{\mathcal{Y}}, \mathbf{t}^i_{nn}, \mathcal{Z}^i_{nn}, \mathcal{Z}^i_{\mathcal{Y}}, \mathbf{x}^i_{sy}, \mathbf{x}^i_{nn}) \in \mathcal{S}$. Suppose there exists a latent minimizer,*

$$\mathbf{z}^* \in \arg\min_{\hat{\mathbf{z}} \in \mathcal{Z}^i_{\mathcal{Y}}} g_{sy}((\mathbf{t}^i_{\mathcal{Y}}, \hat{\mathbf{z}}), \mathbf{x}^i_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})),$$

such that $E((\mathbf{t}^i_{\mathcal{Y}}, \mathbf{z}^*), \mathbf{x}^i_{sy}, \mathbf{x}^i_{nn}, \mathbf{w}_{sy}, \mathbf{w}_{nn})$ is finite.

If the latent optimal value-function, V, is differentiable with respect to every output of the neural model, $\mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})$, then the gradient of V with respect to $\mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})$ is:

$$\nabla_{\mathbf{g}_{nn}(\mathbf{x}^i_{nn},\mathbf{w}_{nn})} V(\mathbf{t}^i_{\mathcal{Y}}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})) \quad (14)$$
$$= \nabla_4 g_{sy}((\mathbf{t}^i_{\mathcal{Y}}, \mathbf{z}^*), \mathbf{x}^i_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})),$$

where $\nabla_4 g_{sy}$ denotes the gradient of the symbolic component with respect to its $4'th$ argument with all other arguments fixed at the specified values.

Similarly, if V is differentiable with respect to every component of the symbolic weight, $\mathbf{w}_{sy}$, then the gradient of V with respect to $\mathbf{w}_{sy}$ is:

$$\nabla_{\mathbf{w}_{sy}} V(\mathbf{t}^i_{\mathcal{Y}}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})) \quad (15)$$
$$= \nabla_3 g_{sy}((\mathbf{t}^i_{\mathcal{Y}}, \mathbf{z}^*), \mathbf{x}^i_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})).$$

Sketch of Proof. For an arbitrary index $i \in \{1, \cdots, d_{nn}\}$, let $\mathbf{e}^i$ be the $i'th$ standard basis vector of $\mathbb{R}^{d_{nn}}$. For any $\delta \in \mathbb{R}$, by (11) we have

$$g_{sy}((\mathbf{t}^i_{\mathcal{Y}}, \mathbf{z}^*), \mathbf{x}^i_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn}) + \delta\mathbf{e}^i)$$
$$- g_{sy}((\mathbf{t}^i_{\mathcal{Y}}, \mathbf{z}^*), \mathbf{x}^i_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn}))$$
$$\geq V(\mathbf{t}^i_{\mathcal{Y}}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn}) + \delta\mathbf{e}^i)$$
$$- V(\mathbf{t}^i_{\mathcal{Y}}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}^i_{nn}, \mathbf{w}_{nn})).$$

For $\delta \neq 0$, dividing both sides by δ and taking the limit as $\delta \to 0+$ and as $\delta \to 0-$ yields upper and lower bounds relating partial derivatives of g_{sy} to V when V is right and left hand differentiable, respectively. Then, by the squeeze theorem, we obtain the partial derivatives implied by the gradient in (14) when V is differentiable with respect to the neural outputs. A similar approach is used to obtain gradients with respect to symbolic weights in (15). □

Thm. 5.1 states *if* the value-function is differentiable, then the gradients have the form provided in (14) and (15). Milgrom and Segal (2002) also provide sufficient conditions for guaranteeing differentiability of optimal value-functions. Beyond Milgrom and Segal's (2002) work, there is extensive literature on analyzing the sensitivity of optimal value-functions and guaranteeing their differentiability, including the seminal papers of (Danskin 1966) on parameterized objective functions and (Rockafellar 1974) for parameterized constraints. We direct the reader to the cited articles for properties that guarantee differentiability of value-functions and, hence, NeSy-EBM value-based losses. The conditions ensuring differentiability of the optimal value-functions as well as the tractability of computing the gradient of the symbolic component with respect to its arguments in (14) and (15) directly connect to the energy function architecture and modeling patterns discussed in the previous section. Specifically, if principled gradient-based learning is desired, then practitioners must design the symbolic potential such that it is 1) differentiable with respect to the neural output and symbolic potentials, 2) the gradient of the symbolic potential with respect to its arguments is tractable, and 3) it satisfies sufficient conditions for ensuring differentiability of its minimizing value over the targets.

An analogous theorem can be constructed for similarly defined value-functions and hence used for value-based losses in general. In our empirical analysis, we use the gradients suggested by the theorem to fit neural and symbolic weights to a cross-entropy neural loss and an energy value-based loss with binary target variable domains.

6 Empirical Analysis

We analyze the NeSy modeling patterns and learning losses on three unique inference tasks, each showcasing compelling applications of NeSy-EBMs. The experiments are designed to answer the following research questions:

1. RQ1: Does integrating neural and symbolic systems improve the performance of deep models?
2. RQ2: Can DSVar and DSPar modeling patterns be used with a value-based loss for semi-supervised learning?
3. RQ3: Can a neural component be trained as a weighting function for a symbolic mixture of experts model?
4. RQ4: Can LLMs generate symbolic potentials for the DSPot modeling pattern to improve its reasoning?

All experiments are implemented using the open-source neural probabilistic soft logic (NeuPSL) (Pryor et al. 2023). A NeuPSL program is a collection of weighted first-order logical clauses and linear arithmetic inequalities referred to as *rules*. The rules are translated into a symbolic potential using Łukasiewicz real-valued logic semantics and, therefore, measures the total weighted satisfaction of variables. The NeuPSL symbolic component is a non-smooth convex function, and the targets may be continuous in $[0, 1]$ or Boolean in $\{0, 1\}$. Thus, prediction is a mixed integer program we pass to Gurobi to solve. [1]

6.1 Semi-Supervised Learning

We begin by exploring the prediction performance of DSVar and DSPar models on two variations of *Visual Sudoku*, a standard evaluation setting in the NeSy literature (Wang et al. 2019; Augustine et al. 2022). Here, Sudoku puzzles are 9×9 grids of MNIST images related by the rules of Sudoku: no duplicate digits in any row, column, or block. We propose two semi-supervised NeSy learning settings:

- *VS-Complete*: Valid and complete puzzles.
- *VS-Solving*: Incomplete puzzles with 30 clues.

Sets of $(32, 25, 50)$ and $(64, 50, 100)$ are provided as training, validation, and testing for VS-Complete and VS-Solving, respectively. In both settings, the task is to classify every MNIST digit in the puzzle. The proportion of labeled digits in the training data is varied over $\lambda \in \{1.0, 0.1, 0.05, 0.01\}$. In other words, if n_{digits} exist in the training data across all the provided puzzles, $\lambda \cdot n_{digits}$ are available for supervised training of the neural component using the cross-entropy neural loss. The remaining $(1 - \lambda) \cdot n_{digits}$ are unlabeled. We evaluate the performance of the following models.

- *ResNet18*: A baseline ResNet18 model (He et al. 2016).
- *DSVar*: A deep symbolic variables NeSy-EBM. (illustrated in Fig. 2 for VS-Solving).
- *DSPar*: A deep symbolic parameters NeSy-EBM. Neural component predictions are used as a prior.

The target variables represent the categorical digit label of every entry in the puzzle for both the DSVar and DSPar models. Further, for the DSVar model, the targets corresponding to non-empty cells are assigned to the neural predictions, as formalized in (4). The neural component of the DSVar and DSPar models is the baseline ResNet18 digit classifier. The NeuPSL rules for instantiating the DSVar and DSPar symbolic components are the same. Further, the weights of the DSVar and DSPar are trained using the energy loss (12).

The test set digit accuracy of both the neural and symbolic components of the NeSy-EBMs and at all levels of superivsion is provided in Tab. 1. Overall, the results indicate that both the DSVar and DSPar neural components outperform the ResNet18 model at every level of supervision, providing evidence for an affirmative answer to RQ1 and RQ2. In *VS-Solving*, the DSPar neural component consistently outperforms the DSVar model. Further, we see that the DSPar model's symbolic component achieves the highest accuracy in every experiment. Thus, the DSPar model's symbolic component is able to resolve errors made by the neural component during training using the Sudoku rules.

6.2 Mixture of Symbolic Experts

In this experiment, we provide an additional motivating use case for the deep symbolic parameter modeling pattern. We use the neural model as a weighting function for the symbolic component, which is a mixture of experts. We evaluate this model on a synthetic node classification problem, which closely mirrors Ex. 4.2. This graph comprises disjoint, fully connected vertex and edge subgraphs with each subgraph either having: 1) nodes with random labels but representative features, or 2) nodes with a common label but random features. Experiments are performed in three inductive settings, where new nodes are added to pre-existing subgraphs with the following information concatenated to the node features:

- $G + OH$: A sample from a Gaussian conditioned on the label and a one-hot encoding of the subgraph id.
- $G{+}G$: A sample from Gaussians conditioned on the label and on the subgraph id.

The number of subgraphs is 25 with minimum and maximum sizes of 10 and 15, and the node label space consists of 4 labels. An equal number of subgraphs are generated from the symbolic structures each containing at least two nodes in the train set. Each experiment was performed on 5 splits using $60/30/10$ train-test-valid partitions. Experiments are conducted using the following models:

- *Multi-Layer Perception (MLP).*
- *PSL (Bach et al. 2017):* A baseline symbolic model.
- *DSPar:* A deep symbolic parameters NeSy-EBM.

The symbolic components of both the PSL and DSPar models are the same and consist of the two constraints in Ex. 4.2 and Fig. 3. Further, the neural component of the DSPar

[1] Code and data: https://github.com/linqs/aaai-make24.

	Labeled: λ	Neural: $\mathbf{g}_{nn}$ ResNet18	DSVar	DSPar	Symbolic: $\mathbf{g}_{sy}$ DSPar
VS-Complete	1.00	98.00 ± 0.18	98.14 ± 0.36	98.07 ± 0.19	$\mathbf{100.00 \pm 0.00}$
	0.10	92.44 ± 0.78	97.55 ± 0.26	97.47 ± 0.36	$\mathbf{100.00 \pm 0.00}$
	0.05	83.35 ± 2.39	97.36 ± 0.11	97.35 ± 0.42	$\mathbf{100.00 \pm 0.00}$
	0.01	49.75 ± 3.40	30.95 ± 7.94	62.62 ± 19.48	$\mathbf{64.44 \pm 19.89}$
VS-Solving	1.00	98.10 ± 0.13	98.02 ± 0.19	98.09 ± 0.25	$\mathbf{99.40 \pm 0.01}$
	0.10	88.98 ± 1.60	93.53 ± 0.39	95.13 ± 0.79	$\mathbf{97.62 \pm 0.60}$
	0.05	81.89 ± 2.38	93.39 ± 1.03	93.95 ± 0.17	$\mathbf{97.12 \pm 0.26}$
	0.01	43.62 ± 3.49	70.87 ± 15.38	79.97 ± 9.27	$\mathbf{81.19 \pm 10.01}$

Table 1: Digit accuracy on semi-supervised visual Sudoku.

model is two MLPs that predict the constraint weights. The weights of the PSL and DSPar models are trained using the bilevel NeSy learning framework introduced by Dickens et al. (2024) with a cross-entropy loss on the prediction.

	MLP	PSL	DSPar
$G + OH$	88.10 ± 5.69	87.06 ± 5.61	**100.00 ± 0.00**
$G + G$	89.04 ± 2.72	86.71 ± 5.46	**93.35 ± 2.23**

Table 2: Accuracy on citation network node classification.

Tab. 2 reports the average and standard deviation of the categorical accuracy of each model. In all cases, *DSPar* performs the best, only losing a few percentage points with the most challenging features ($G + G$). Notably, *PSL* cannot model both symbolic structures of the dataset, motivating the use of the mixture of symbolic experts approach.

6.3 Enhancing Large Language Model Reasoning

Finally, we explore the integration of LLMs with symbolic solvers to improve problem-solving and reasoning. Specifically, we study the predictive performance of the deep symbolic potentials modeling pattern proposed in Section 4.3 in the following logical reasoning dataset.

- *Logical Deduction* (Srivastava et al. 2022): Multiple-choice questions that require deducing the order of a sequence of objects given a natural language description.

A prompt with two examples is provided during inference. The test set consists of 300 logical deduction problems. We compare our predictive performance with the baseline and models presented in Pan et al. (2023) on gpt-3.5-turbo and gpt-4 (Achiam et al. 2023).

- *LLMs (Standard)*: LLM directly answers query.
- *Chain-of-Thought (CoT)* (Wei et al. 2022): LLM is prompted for a step-by-step explanation with the answer.
- *Logic-LM* (Pan et al. 2023): A DSPot NeSy-EBM. The symbolic component is a *python-constraint* program .
- *NeuPSL*: A DSPot NeSy-EBM. An LLM is used to generate NeuPSL rules.

Similar to Pan et al. (2023), models are tested using identical in-context examples. For reproducibility, the temperature parameter is fixed at 0, so generated responses are with the highest likelihood.

	Standard	CoT	DSPot Logic-LM	DSPot NeuPSL
GPT-3.5	40.00	42.33	65.67	**70.33**
GPT-4	71.33	75.25	87.63	**90.67**

Table 3: Accuracy on logical deduction question answering.

Tab. 3 reports the accuracy of each model. In all cases, both DSPot models outperform the standard LLM and CoT models. Additionally, the LLMs produce feasible symbolic potentials for DSPot-NeuPSL 89.0% and 98.7% of the time with gpt-3.5-turbo and gpt-4, respectively.

7 Limitations

We acknowledge our modeling pattern taxonomy is not exhaustive. For instance, we do not cover NeSy systems designed for extracting symbolic knowledge from deep neural networks as in (d'Avila Garcez et al. 2019). Additionally, we focus on empowering LLMs with commonsense reasoning and domain knowledge for question answering and do not explore their applications in other complex reasoning tasks summarization, or explanation.

8 Conclusion and Future Work

Our general taxonomy of NeSy modeling patterns is a new tool for understanding and bridging NeSy challenges to the broader AI literature. Furthermore, the empirical analysis demonstrated multiple compelling applications of the modeling, including empowering LLMs, as shown with notable improvements to GPT-4's reasoning capabilities, underscoring the practical relevance of NeSy. A promising direction for future work is a comprehensive loss categorization with an extended empirical analysis. Additionally, we advocate for research that explores new NeSy applications that may require complex combinations of modeling patterns.

Acknowledgments

This work was partially supported by the National Science Foundation grant CCF-2023495.

References

Achiam, O. J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; Avila, R.; and et al. 2023. GPT-4 Technical Report.

Ahmed, K.; Teso, S.; Chang, K.-W.; Van den Broeck, G.; and Vergari, A. 2022. Semantic Probabilistic Layers for Neuro-Symbolic Learning. In *NeurIPS*.

Augustine, E.; Pryor, C.; Dickens, C.; Pujara, J.; Wang, W. Y.; and Getoor, L. 2022. Visual Sudoku Puzzle Classification: A Suite of Collective Neuro-Symbolic Tasks. In *International Workshop on Neural-Symbolic Learning and Reasoning (NeSy)*.

Bach, S.; Broecheler, M.; Huang, B.; and Getoor, L. 2017. Hinge-Loss Markov Random Fields and Probabilistic Soft Logic. *Journal of Machine Learning Research (JMLR)*, 18(1): 1–67.

Bader, S.; and Hitzler, P. 2005. Dimensions of Neural-symbolic Integration - A Structured Survey. *ArXiv*.

Badreddine, S.; d'Avila Garcez, A.; Serafini, L.; and Spranger, M. 2022. Logic Tensor Networks. *Artificial Intelligence (AI)*, 303(4): 103649.

Besold, T. R.; d'Avila Garcez, A. S.; Bader, S.; Bowman, H.; Domingos, P. M.; Hitzler, P.; Kühnberger, K.; Lamb, L. C.; Lowd, D.; Lima, P. M. V.; de Penning, L.; Pinkas, G.; Poon, H.; and Zaverucha, G. 2022. Neural-Symbolic Learning and Reasoning: A Survey and Interpretation. *Neuro-Symbolic Artificial Intelligence: The State of the Art*.

Cornelio, C.; Stuehmer, J.; Xu Hu, S.; and Hospedales, T. 2023. Learning Where and When to Reason In Neuro-Symbolic Inference. In *ICLR*.

Danskin, J. 1966. The Theory of Max-Min, with Applications. *SIAM Journal on Applied Mathematics*, 14(4): 641–664.

Dash, T.; Chitlangia, S.; Ahuja, A.; and Srinivasan, A. 2022. A Review of Some Techniques for Inclusion of Domain-Knowledge into Deep Neural Networks. *Scientific Reports*, 12(1): 1040.

d'Avila Garcez, A.; Gori, M.; Lamb, L. C.; Serafini, L.; Spranger, M.; and Tran, S. N. 2019. Neural-Symbolic Computing: An Effective Methodology for Principled Integration of Machine Learning and Reasoning. *Journal of Applied Logics*, 6(4): 611–632.

d'Avila Garcez, A. S.; Broda, K.; and Gabbay, D. M. 2002. *Neural-Symbolic Learning Systems: Foundations and Applications*. Springer.

d'Avila Garcez, A. S.; Lamb, L. C.; and Gabbay, D. M. 2009. *Neural-Symbolic Cognitive Reasoning*. Springer.

De Raedt, L.; Dumančić, S.; Manhaeve, R.; and Marra, G. 2020. From Statistical Relational to Neuro-Symbolic Artificial Intelligence. In *IJCAI*.

Dickens, C.; Gao, C.; Pryor, C.; Wright, S.; and Getoor, L. 2024. Convex and Bilevel Optimization for Neuro-Symbolic Inference and Learning. *ArXiv*.

Evans, R.; and Grefenstette, E. 2018. Learning Explanatory Rules from Noisy Data. *JAIR*, 61: 1–64.

Giunchiglia, E.; Stoian, M. C.; and Lukasiewicz, T. 2022. Deep Learning with Logical Constraints. In *IJCAI*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

Lamb, L. C.; d'Avila Garcez, A.; Gori, M.; Prates, M. O. R.; Avelar, P. H. C.; and Vardi, M. Y. 2020. Graph Neural Networks Meet Neural-Symbolic Computing: A Survey and Perspective. In *IJCAI*.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

LeCun, Y.; Chopra, S.; Hadsell, R.; Ranzato, M.; and Huang, F. J. 2006. A Tutorial on Energy-Based Learning. *Predicting Structured Data*, 1(0).

Manhaeve, R.; Dumančić, S.; Kimmig, A.; Demeester, T.; and De Raedt, L. 2021. Neural Probabilistic Logic Programming in DeepProbLog. *Artificial Intelligence (AI)*, 298: 103504.

Marconato, E.; Teso, S.; Vergari, A.; and Passerini, A. 2023. Not All Neuro-Symbolic Concepts Are Created Equal: Analysis and Mitigation of Reasoning Shortcuts. In *NeurIPS*.

Milgrom, P.; and Segal, I. 2002. Envelope Theorems for Arbitrary Choice Sets. *Econometrica*, 70(2): 583–601.

Pan, L.; Albalak, A.; Wang, X.; and Wang, W. Y. 2023. Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning. *ArXiv*.

Pryor, C.; Dickens, C.; Augustine, E.; Albalak, A.; Wang, W. Y.; and Getoor, L. 2023. NeuPSL: Neural Probabilistic Soft Logic. In *IJCAI*.

Rockafellar, R. 1974. Conjugate Duality and Optimization. In *Regional Conference Series in Applied Mathematics*.

Srivastava, A.; Rastogi, A.; Rao, A.; Shoeb, A. A. M.; Abid, A.; Fisch, A.; Brown, A. R.; Santoro, A.; Gupta, A.; Garriga-Alonso, A.; Kluska, A.; Lewkowycz, A.; Agarwal, A.; Power, A.; Ray, A.; Warstadt, A.; Kocurek, A. W.; Safaya, A.; Tazarv, A.; Xiang, A.; Parrish, A.; Nie, A.; Hussain, A.; Askell, A.; Dsouza, A.; Slone, A.; Rahane, A. A.; Iyer, A. S.; Andreassen, A.; Madotto, A.; Santilli, A.; Stuhlmuller, A.; Dai, A. M.; La, A.; Lampinen, A. K.; Zou, A.; Jiang, A.; Chen, A.; Vuong, A.; Gupta, A.; Gottardi, A.; Norelli, A.; Venkatesh, A.; Gholamidavoodi, A.; Tabassum, A.; Menezes, A.; Kirubarajan, A.; Mullokandov, A.; Sabharwal, A.; Herrick, A.; Efrat, A.; Erdem, A.; Karakacs, A.; and et al. 2022. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. *ArXiv*.

van Krieken, E.; Acar, E.; and van Harmelen, F. 2022. Analyzing Differentiable Fuzzy Logic Operators. *Artificial Intelligence (AI)*, 302: 103602.

Wang, P.; Donti, P.; Wilder, B.; and Kolter, Z. 2019. Satnet: Bridging Deep Learning and Logical Reasoning Using a Differentiable Satisfiability Solver. In *ICML*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; hsin Chi, E. H.; Xia, F.; Le, Q.; and Zhou, D. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. *ArXiv*.

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 *ICML*.

Yang, F.; Yang, Z.; and Cohen, W. W. 2017. Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In *NeurIPS*.

Yang, Z.; Ishay, A.; and Lee, J. 2020. NeurASP: Embracing Neural Networks into Answer Set Programming. In *IJCAI*.