NeuPSL: Neural Probabilistic Soft Logic

Connor Pryor[1] , **Charles Dickens**[1] , **Eriq Augustine** [1] , **Alon Albalak** [2] ,
William Yang Wang [2] and **Lise Getoor** [1]
[1] UC Santa Cruz
[2] UC Santa Barbara
cfpryor@ucsc.edu, cadicken@ucsc.edu, eaugusti@ucsc.edu, alon_albalak@ucsb.edu,
william@cs.ucsb.edu, getoor@ucsc.edu

Abstract

In this paper, we introduce *Neural Probabilistic Soft Logic* (NeuPSL), a novel neuro-symbolic (NeSy) framework that unites state-of-the-art symbolic reasoning with the low-level perception of deep neural networks. To model the boundary between neural and symbolic representations, we propose a family of energy-based models, *NeSy Energy-Based Models*, and show that they are general enough to include NeuPSL and many other NeSy approaches. Using this framework, we show how to seamlessly integrate neural and symbolic parameter learning and inference in NeuPSL. Through an extensive empirical evaluation, we demonstrate the benefits of using NeSy methods, achieving upwards of 30% improvement over independent neural network models. On a well-established NeSy task, MNIST-Addition, NeuPSL demonstrates its joint reasoning capabilities by outperforming existing NeSy approaches by up to 10% in low-data settings. Furthermore, NeuPSL achieves a 5% boost in performance over state-of-the-art NeSy methods in a canonical citation network task with up to a 40 times speed up.

1 Introduction

The field of artificial intelligence (AI) has long sought a symbiotic union of neural and symbolic methods. Neural-based methods excel at low-level perception and learn from large training data sets but struggle with interpretability and generalizing in low-data settings. Meanwhile, symbolic methods can effectively use domain knowledge, context, and common sense to reason with limited data but have difficulty representing complex low-level patterns. Recently, neuro-symbolic computing (NeSy) [Besold *et al.*, 2017; d'Avila Garcez *et al.*, 2019; De Raedt *et al.*, 2020] has emerged as a promising new research area with the goal of developing systems that integrate neural and symbolic methods in a mutually beneficial manner.

A neural and symbolic union has the potential to yield two highly desirable capabilities - the ability to perform structured prediction (*joint inference*) across related examples that possess complex low-level features and the ability to jointly learn (*joint learning*) and adapt parameters over neural and symbolic models simultaneously. For instance, predicting the result of competitions between teams using historical performance statistics in a tournament bracket requires methods to perform joint inference to reason over low-level trends and avoid inconsistencies such as two first-place finishes. Unfortunately, joint inference problems quickly grow in complexity as the output space typically increases combinatorially. For example, in the tournament setting, as the number of entries increases, the number of potential solutions grows exponentially ($O(2^n)$). An open challenge in the NeSy community is scaling joint inference and reasoning.

This paper introduces *Neural Probabilistic Soft Logic* (NeuPSL), a novel NeSy method that integrates deep neural networks with a symbolic method designed for fast joint learning and inference. NeuPSL extends probabilistic soft logic (PSL) [Bach *et al.*, 2017], a state-of-the-art and scalable probabilistic programming framework that can reason statistically (using probabilistic inference) and logically (using soft rules). PSL has been shown to excel in a wide variety of tasks, including natural language processing [Beltagy *et al.*, 2014; Deng and Wiebe, 2015; Liu *et al.*, 2016; Rospocher, 2018], data mining [Alshukaili *et al.*, 2016; Kimmig *et al.*, 2019], recommender systems [Kouki *et al.*, 2015], knowledge graph discovery [Pujara *et al.*, 2013], fairness modeling [Farnadi *et al.*, 2019; Dickens *et al.*, 2020], and causal reasoning [Sridhar *et al.*, 2018]. The key innovation of NeuPSL is a new class of predicates that rely on neural network output for their values. This change fundamentally alters the learning and joint inference problems by requiring efficient integrated symbolic and neural parameter learning. The appeal of this extension is that it allows for the semantics and implementation of the symbolic language to remain the same as PSL, while also incorporating the added benefit of low-level neural perception. To gain a deeper understanding of optimizing the symbolic and neural parameters, we propose a versatile mathematical framework, Neuro-Symbolic Energy-Based Models (NeSy-EBMs), that enables many NeSy systems to utilize established Energy-Based Model learning losses and algorithms. Utilizing this theory and leveraging the unique relaxation properties of PSL, we show that a gradient over these neural predicates can be calculated and passed back to common back-propagation engines such as PyTorch or Tensorflow, allowing for scalable end-to-end gradient training.

Our key contributions include: 1) We define *Neuro-Symbolic Energy-Based Models* (NeSy-EBMs), a family of energy-based models, and show how they provide a foundation for describing, understanding and comparing NeSy systems. 2) We introduce NeuPSL, describe how it fits into the NeSy ecosystem and supports scalable joint inference, and show how it can be trained end-to-end using a joint energy-based learning loss. 3) We perform extensive evaluations over two image classification tasks and two citation network datasets. Our results show NeuPSL consistently outperforms existing approaches on joint inference tasks and can more efficiently leverage structure, particularly in low-data settings.

2 Related Work

Neuro-symbolic computing (NeSy) is an active area of research that aims to incorporate logic-based reasoning with neural networks [d'Avila Garcez *et al.*, 2002; Bader and Hitzler, 2005; d'Avila Garcez *et al.*, 2009; Serafini and d'Avila Garcez, 2016; Besold *et al.*, 2017; Donadello *et al.*, 2017; Yang *et al.*, 2017; Evans and Grefenstette, 2018; Manhaeve *et al.*, 2021; d'Avila Garcez *et al.*, 2019; De Raedt *et al.*, 2020; Lamb *et al.*, 2020; Badreddine *et al.*, 2022]. The advantages of NeSy systems include interpretability, robustness, and the ability to integrate various sub-problem solutions (such as perception, reasoning, and decision-making). For a thorough introduction to NeSy literature, we refer the reader to the excellent surveys by Besold *et al.* (2017) and De Raedt *et al.* (2020). In this section, we identify key NeSy research categories and provide a brief description of each.

Differentiable frameworks of logical reasoning: Methods in this category use neural networks' universal function approximation properties to emulate logical reasoning inside networks. Examples include: Rocktäschel and Riedel (2017), Bošnjak *et al.* (2017), Evans and Grefenstette (2018), and Cohen *et al.* (2020).

Constrained Output: These approaches enforce constraints or regularizations on the output of neural networks. Examples include: Hu *et al.* (2016), Diligenti *et al.* (2017), Donadello *et al.* (2017), Mehta *et al.* (2018), Xu *et al.* (2018), and Nandwani *et al.* (2019).

Executable logic programs: These approaches use neural models to build executable logical programs. Examples include Liang *et al.* (2017) and Mao *et al.* (2019). We highlight Logic Tensor Networks (LTNs) [Badreddine *et al.*, 2022], as we include this approach in our empirical evaluation. LTNs connect neural predictions into functions representing symbolic relations with real-valued or fuzzy logic semantics.

Neural networks as predicates: This line of work integrates neural networks and probabilistic reasoning by introducing neural networks as predicates in the logical formulae. This technique provides a very general and flexible framework for NeSy reasoning and allows for the use of multiple networks as well as the full incorporation of constraints and relational information. Examples include DASL [Sikka *et al.*, 2020], NeurASP [Yang *et al.*, 2020], Nuts&Bolts [Sachan *et al.*, 2018], DeepProbLog (DPL) [Manhaeve *et al.*, 2021], and our proposed method (Neural Probabilistic Soft Logic). DPL combines general-purpose neural networks with the probabilistic modeling of ProbLog [De Raedt *et al.*, 2007] in a way that allows for learning and inference over complex tasks, such as program induction. We include DPL in our empirical evaluation.

3 Neuro-Symbolic Energy-Based Models

With the success and growth of NeSy research, there is an increasing need for a common formalization of NeSy systems to accelerate the research and understanding of the field. We fill this need with a general mathematical framework, *Neuro-Symbolic Energy-Based Models* (NeSy-EBMs). NeSy-EBMs encompass previous approaches and establishes the foundation of our approach. Energy-Based Models (EBMs) [LeCun *et al.*, 2006] measure the compatibility of a collection of observed (or input) variables $\mathbf{x} \in \mathcal{X}$ and target (or output) variables $\mathbf{y} \in \mathcal{Y}$ with a scalar-valued *energy function*: $E : \mathcal{Y} \times \mathcal{X} \to \mathbb{R}$. Low energy states of the variables represent high compatibility. Prediction or *inference* in EBMs is performed by finding the lowest energy state of the variables $\mathbf{y}$ given $\mathbf{x}$. Energy functions are parameterized by variables $\mathbf{w} \in \mathcal{W}$, and *learning* is the task of finding a parameter setting that associates low energy to correct solutions.

Building on the well-known EBM framework, NeSy-EBMs are a family of EBMs that integrate neural architectures with explicit encodings of symbolic relations. The input variables are organized into neural, $\mathbf{x}_{nn} \in \mathcal{X}_{nn}$, and symbolic, $\mathbf{x}_{sy} \in \mathcal{X}_{sy}$, vectors. Furthermore, the parameters of the energy function, $\mathbf{w}$, are partitioned into neural weights, $\mathbf{w}_{nn} \in \mathcal{W}_{nn}$, and symbolic weights, $\mathbf{w}_{sy} \in \mathcal{W}_{sy}$. Formally,

Definition 1 (NeSy-EBM)**.** *Let $\mathbf{y} \in \mathcal{Y}$ and $\mathbf{x}_{sy} \in \mathcal{X}_{sy}$ be vectors of variables with symbolic interpretations. Let $\mathbf{g}_{nn}$ be neural networks with **neural weights** $\mathbf{w}_{nn} \in \mathcal{W}_{nn}$ and inputs $\mathbf{x}_{nn} \in \mathcal{X}_{nn}$. A **symbolic potential** is a function of $\mathbf{y}$, $\mathbf{x}_{sy}$, and $\mathbf{g}_{nn}(\cdot)$ parameterized by **symbolic weights** $\mathbf{w}_{sy} \in \mathcal{W}_{sy}$: $\psi(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})) \in \mathbb{R}$. A **NeSy-EBM energy function** is a mapping of a vector of m symbolic potential outputs, $\Psi(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn}) = [\psi_i(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn}))]_{i=1}^{m}$, to a real value: $E(\Psi(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn})) \in \mathbb{R}$.*

NeSy-EBMs are differentiated from one another by the instantiation process, the form of the symbolic potentials, and the definition of the energy function. In appendix, we formally show how two NeSy systems DeepProbLog (DPL) [Manhaeve *et al.*, 2018] and Logic Tensor Networks (LTNs) [Badreddine *et al.*, 2022] fit into the NeSy-EBM framework. In summary, DPL uses neural network outputs to specify event probabilities that are used in logical formulae defining probabilistic dependencies. The definition of the DPL symbolic potentials and energy function are tied to the inference task; a different definition of the symbolic potential and energy function is used to implement marginal versus MAP inference. For marginal, the most common DPL inference, symbolic potentials are functions of marginal probabilities, and the energy function is a joint distribution that is the sum of the symbolic potentials. LTNs instantiate a model which forwards neural network predictions into functions representing symbolic relations with real-valued or fuzzy logic seman-

tics. The fuzzy logic functions are symbolic potentials that are aggregated to define the energy function. The following section will introduce how our approach, NeuPSL, is instantiated as a NeSy-EBM. Using this common framework, understanding and theoretical advances can be made across NeSy approaches.

3.1 Joint Reasoning in NeSy-EBMs

We highlight two important categories of NeSy-EBM energy functions: *joint* and *independent*. Formally, an energy function that is additively separable over the output variables $\mathbf{y}$ is an *independent energy function*, i.e., corresponding to each of the n_y components of the output variable $\mathbf{y}$ there exists functions n_y functions $E_1(\mathbf{y}[1], \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}(\mathbf{x}_{nn}, \mathbf{w}_{nn}))$, $\cdots$, $E_{n_y}(\mathbf{y}[n_y], \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}(\mathbf{x}_{nn}, \mathbf{w}_{nn}))$ such that

$$E(\cdot) = \sum_{i=1}^{n_y} E_i(\mathbf{y}[i], \mathbf{x}_{sy}, \mathbf{w}_{sy}, \mathbf{g}(\mathbf{x}_{nn}, \mathbf{w}_{nn})).$$

While a function that is not separable over output variables $\mathbf{y}$ is a *joint energy function*. This categorization allows for an important distinction during inference and learning. Independent energy functions simplify inference and learning as finding an energy minimizer, $\mathbf{y}^*$, can be distributed across the independent functions E_i. In other words, the predicted value for a variable $\mathbf{y}[i]$ has no influence over that of $\mathbf{y}[j]$ where $j \neq i$ and can therefore be predicted separately, i.e., independently. However, independent energy functions cannot leverage some joint information that may be used to improve predictions. See appendix for further details.

4 Neural Probabilistic Soft Logic

Having laid the NeSy-EBM groundwork, we now introduce *Neural Probabilistic Soft Logic* (NeuPSL), a novel NeSy-EBM framework that extends the probabilistic soft logic (PSL) framework [Bach *et al.*, 2017]. At its core, NeuPSL leverages the power of neural networks' low-level perception by seamlessly integrating their outputs with a collection of symbolic potentials generated through a PSL program. Figure 1 provides a graphical representation of this process. The symbolic potentials and neural networks together define a *deep hinge-loss Markov random field* (*Deep-HL-MRF*), a tractable probabilistic graphical model that supports scalable convex joint inference. This section provides a comprehensive description of how NeuPSL instantiates its symbolic potentials and how the symbolic potentials are combined to define an energy function, while the following section details NeuPSL's end-to-end neural-symbolic inference, learning, and joint reasoning processes.

NeuPSL instantiates the symbolic potentials of its energy function using the PSL language where dependencies between relations and attributes of entities in a domain, defined as *atoms*, are encoded with weighted first-order logical clauses and linear arithmetic inequalities referred to as *rules*. To illustrate, consider a setting in which a neural network is used to classify the species of an animal in an image. Further, suppose there exists external information suggesting when two images may contain the same entity. The information linking the images may come from various sources, such as the images' caption or metadata indicating the images were captured by the same device within a short period of time. NeuPSL represents the neural network's animal classification of an image ($\texttt{Image}_1$) as a species ($\texttt{Species}$) with the atom $\textsc{Neural}(\texttt{Image}_1, \texttt{Species})$ and the probability that two images ($\texttt{Image}_1$ and $\texttt{Image}_2$) contain the same entity with the atom $\textsc{SameEntity}(\texttt{Image}_1, \texttt{Image}_2)$. Additionally, we represent NeuPSL's classification of $\texttt{Image}_2$ with $\textsc{Class}(\texttt{Image}_2, \texttt{Species})$. The following weighted logical rule in NeuPSL represents the notion that two images identified as the same entity may also be of the same species:

$$\begin{aligned} w : \textsc{Neural}(\texttt{Image}_1, \texttt{Species}) \\ \wedge \textsc{SameEntity}(\texttt{Image}_1, \texttt{Image}_2) \\ \rightarrow \textsc{Class}(\texttt{Image}_2, \texttt{Species}) \end{aligned} \quad (1)$$

The parameter w is the weight of the rule, and it quantifies its relative importance in the model. Note these rules can either be hard or soft constraints. Atoms and weighted rules are templates for creating symbolic potentials or soft constraints. To create these symbolic potentials, atoms and rules are instantiated with observed data and neural predictions. Atoms instantiated with elements from the data are referred to as *ground atoms*. Then, valid combinations of ground atoms substituted in the rules create *ground rules*. To illustrate, suppose that there are two images $\{Id1, Id2\}$ and three species classes $\{Cat, Dog, Frog\}$. Using the above data for cats would result in the following ground rules (analogous ground rules would be created for dogs and frogs):

$$\begin{aligned} w : \textsc{Neural}(Id1, Cat) \wedge \textsc{SameEntity}(Id1, Id2) \\ \rightarrow \textsc{Class}(Id2, Cat) \\ w : \textsc{Neural}(Id2, Cat) \wedge \textsc{SameEntity}(Id2, Id1) \\ \rightarrow \textsc{Class}(Id1, Cat) \end{aligned}$$

Ground atoms are mapped to either an observed variable, $x_{sy,i}$, target variable, y_i, or a neural function with inputs $\mathbf{x}_{nn}$ and parameters $\mathbf{w}_{nn,i}$: $g_{nn,i}(\mathbf{x}_{nn}, \mathbf{w}_{nn,i})$. Then, variables are aggregated into the vectors $\mathbf{x}_{sy} = [x_{sy_i}]_{i=1}^{n_x}$ and $\mathbf{y} = [y_i]_{i=1}^{n_y}$ and neural outputs are aggregated into the vector $\mathbf{g}_{nn} = [g_{nn,i}]_{i=1}^{n_g}$. Ground rules are either logical (e.g., Equation 1) or arithmetic defined over $\mathbf{x}_{sy}$, $\mathbf{y}$, and $\mathbf{g}_{nn}$. These ground rules create one or more potentials $\phi(\cdot) \in \mathcal{R}$, where logical rules are relaxed using Łukasiewicz continuous valued logical semantics [Klir and Yuan, 1995]. Each potential $\phi(\cdot)$ is associated with a weight w_{psl} inherited from its instantiating rule. The potentials and weights from the instantiation process are used to define a member of a tractable class of graphical models, *deep hinge-loss Markov random fields* (Deep-HL-MRF):

Definition 2 (Deep Hinge-Loss Markov Random Field)**.** *Let $\mathbf{y} \in [0,1]^{n_y}$ and $\mathbf{x}_{sy} \in [0,1]^{n_x}$ be vectors of $[0,1]$ valued variables. Let $\mathbf{g}_{nn} = [g_{nn,i}]_{i=1}^{n_g}$ be functions with corresponding parameters $\mathbf{w}_{nn} = [\mathbf{w}_{nn,i}]_{i=1}^{n_g}$ and inputs $\mathbf{x}_{nn}$. A **deep hinge-loss potential** is a function of the form*

$$\phi(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn}) = \max(l(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})), 0)^{\alpha} \quad (2)$$

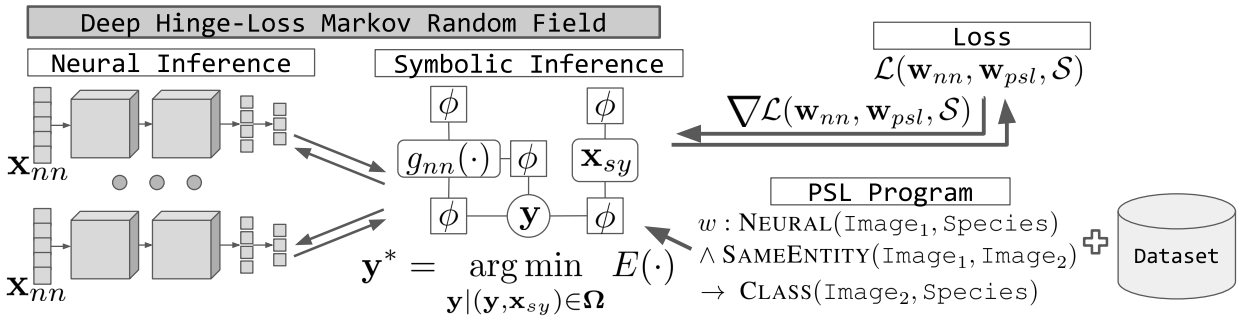

Figure 1: NeuPSL inference and learning pipeline.

*where $l(\cdot)$ is a linear function and $\alpha \in \{1, 2\}$. Let $\mathcal{T} = [t_i]_{i=1}^r$ denote an ordered partition of a set of m deep hinge-loss potentials: $\{\phi_1, \cdots, \phi_m\}$. For each partition t_i define $\Phi_i(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn}) := \sum_{j\in t_i} \phi_i(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn})$ and let $\mathbf{\Phi}(\mathbf{y}, \mathbf{x_{sy}}, \mathbf{x_{nn}}, \mathbf{w_{nn}}) := [\Phi_i(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn})]_{i=1}^r$. Further, let $\mathbf{w}_{psl} = [w_{psl,i}]_{i=1}^r$ be a vector of non-negative weights corresponding to the partition $\mathcal{T}$. Then, a **deep hinge-loss energy function** is*

$$E(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn}, \mathbf{w}_{psl}) = \mathbf{w}_{psl}^T \mathbf{\Phi}(\mathbf{y}, \mathbf{x_{sy}}, \mathbf{x_{nn}}, \mathbf{w_{nn}}) \tag{3}$$

*Further, let $\mathbf{c} = [c_i]_{i=1}^q$ be a vector of q linear constraints in standard form, defining the feasible set $\mathbf{\Omega} = \{\mathbf{y}, \mathbf{x}_{sy} \,|\, c_i(\mathbf{y}, \mathbf{x}_{sy}) \leq 0, \forall i \in \{0, \cdots, q\}\}$. Then a **deep hinge-loss Markov random field**, $\mathcal{P}$, with random variables $\mathbf{y}$ conditioned on $\mathbf{x}_{sy}$ and $\mathbf{x}_{nn}$ is a probability density of the form*

$$P(\mathbf{y}|\mathbf{x}_{sy}, \mathbf{x}_{nn}) = \begin{cases} \frac{\exp(-E(\cdot))}{\int_{\mathbf{y}|\mathbf{y},\mathbf{x}_{sy}\in\mathbf{\Omega}} \exp(-E(\cdot))d\mathbf{y}} & (\mathbf{y}, \mathbf{x}_{sy}) \in \mathbf{\Omega} \\ 0 & o.w. \end{cases}$$

Deep-HL-MRFs naturally fit into the NeSy-EBM framework. The symbolic potentials of deep-HL-MRFs are the aggregated and scaled deep hinge-loss potentials:

$$\begin{aligned} &\psi_{NeuPSL}(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{psl}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})) \\ &= \mathbf{w}_{psl} \Phi(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn}) \end{aligned} \tag{4}$$

Then the energy function is the sum of symbolic potentials:

$$\begin{aligned} &E_{NeuPSL}(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn}, \mathbf{w}_{psl}) \\ &= \sum_{i=1}^{r} \psi_{NeuPSL,i}(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{w}_{psl}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})) \end{aligned} \tag{5}$$

5 NeuPSL Inference and Learning

There is a clear connection between neural and symbolic inference in NeuPSL that allows any neural architecture to interact with symbolic reasoning in a simple and expressive manner. The NeuPSL neural-symbolic interface and inference pipeline is shown in Figure 1. *Neural inference* is computing the output of the neural networks given the input $\mathbf{x}_{nn}$, i.e., computing $g_{nn,i}(\mathbf{x}_{nn}, \mathbf{w}_{nn,i})$ for all i. NeuPSL *symbolic inference* minimizes the energy function over $\mathbf{y}$:

$$\mathbf{y}^* = \underset{\mathbf{y}|(\mathbf{y},\mathbf{x}_{sy})\in\mathbf{\Omega}}{\arg\min}\ E(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn}, \mathbf{w}_{psl}) \tag{6}$$

Note that the hinge-loss potentials are convex in $\mathbf{y}$ and hence, with the common constraint enforcing symbolic parameters to be non-negative, i.e., $\mathbf{w}_{psl} > 0$, the energy function is convex in $\mathbf{y}$. Any scalable convex optimizer can be applied to solve (6). NeuPSL uses the alternating direction method of multipliers [Boyd *et al.*, 2010].

NeuPSL learning is the task of finding both neural and symbolic parameters, i.e., rule weights, that assign low energy to correct values of the output variables and higher energies to incorrect values. Learning objectives are functionals mapping an energy function and a set of training examples $\mathcal{S} = \{(\mathbf{y}_i, \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}) : i = 1, \cdots, P\}$ to a real-valued loss. As the energy function for NeuPSL is parameterized by the neural weights $\mathbf{w}_{nn}$ and symbolic weights $\mathbf{w}_{psl}$, we express the learning objective as a function of $\mathbf{w}_{nn}$, $\mathbf{w}_{psl}$, and $\mathcal{S}$: $\mathcal{L}(\mathcal{S}, \mathbf{w}_{nn}, \mathbf{w}_{psl})$. Learning objectives follow the standard empirical risk minimization framework and are therefore separable over the training examples in $\mathcal{S}$ as a sum of per-sample loss functions $L_i(\mathbf{y}_i, \mathbf{x}_i, \mathbf{x}_{nn,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl})$. Concisely, NeuPSL learning is the following minimization:

$$\begin{aligned} &\underset{\mathbf{w}_{nn},\mathbf{w}_{psl}}{\arg\min}\ \mathcal{L}(\mathbf{w}_{nn}, \mathbf{w}_{psl}, \mathcal{S}) \\ &\quad = \underset{\mathbf{w}_{nn},\mathbf{w}_{psl}}{\arg\min} \sum_{i=1}^{P} L_i(\mathbf{y}_i, \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl}) \end{aligned}$$

In the learning setting, variables $\mathbf{y}_i$ from the training set S are partitioned into vectors $\mathbf{y}_{i,t}$ and $\mathbf{z}_i$. The variables $\mathbf{y}_{i,t}$ represent variables for which there is a corresponding truth value, while $\mathbf{z}_i$ represent latent variables. Without loss of generality, we write $\mathbf{y}_i = (\mathbf{y}_{i,t}, \mathbf{z}_i)$.

There are multiple losses that one could motivate for optimizing the parameters of an EBM. Common losses, including the loss we present in this work, use the following terms:

$$\mathbf{z}_i^* = \underset{\mathbf{z}|((\mathbf{y}_{i,t},\mathbf{z}),\mathbf{x}_{sy,i})\in\mathbf{\Omega}}{\arg\min}\ E((\mathbf{y}_{i,t}, \mathbf{z}), \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl})$$

$$\mathbf{y}_i^* = \underset{\mathbf{y}|(\mathbf{y},\mathbf{x}_{sy,i})\in\mathbf{\Omega}}{\arg\min}\ E(\mathbf{y}, \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl})$$

In words, $\mathbf{z}_i^*$ and $\mathbf{y}_i^*$ are the lowest energy states given $(\mathbf{y}_{i,t}, \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i})$ and $(\mathbf{x}_{sy,i}, \mathbf{x}_{nn,i})$, respectively. A special case of learning is when the per-sample losses are not functions of $\mathbf{z}_i^*$ and $\mathbf{y}_i^*$, and more specifically, the losses do not require any subproblem optimization. We refer to this situation as *constraint learning*. Constraint learning reduces the time required per iteration at the cost of expressivity.

All interesting learning losses for NeuPSL are a composition of the energy function. Thus, a gradient-based learning algorithm will require the following partial derivatives: [1]

$$\frac{\partial E(\cdot)}{\partial \mathbf{w}_{psl}[i]} = \Phi_i(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn})$$
$$\frac{\partial E(\cdot)}{\partial \mathbf{w}_{nn}[i]} = \mathbf{w}_{psl}^T \nabla_{\mathbf{w}_{nn}[i]} \Phi(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn})$$

Continuing with the derivative chain rule and noting the potential can be squared ($\alpha = 2$) or linear ($\alpha = 1$), the potential partial derivative with respect to $\mathbf{w}_{nn}[i]$ is the piece-wise defined function:[1]

$$\frac{\partial \phi(\cdot)}{\partial \mathbf{w}_{nn}[i]} = \begin{cases} \frac{\partial}{\partial \mathbf{g}_{nn}[i]} \phi(\cdot) \cdot \frac{\partial}{\partial \mathbf{w}_{nn}[i]} \mathbf{g}_{nn}[i](\cdot) & \alpha = 1 \\ 2 \cdot \phi(\cdot) \cdot \frac{\partial}{\partial \mathbf{g}_{nn}[i]} \phi(\cdot) \cdot \frac{\partial}{\partial \mathbf{w}_{nn}[i]} \mathbf{g}_{nn}[i](\cdot) & \alpha = 2 \end{cases}$$

$$\frac{\partial \phi(\cdot)}{\partial \mathbf{g}_{nn}[i]} = \begin{cases} 0 & \phi(\cdot) = 0 \\ \frac{\partial}{\partial g_{nn}[i]} l(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn})) & \phi(\cdot) > 0 \end{cases}$$

Since $l(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{g}_{nn}(\mathbf{x}_{nn}, \mathbf{w}_{nn}))$ is a linear function, the partial gradient with respect to $\mathbf{g}_{nn}[i]$ is trivial. With the partial derivatives presented here, standard backpropagation-based algorithms for computing gradients can be applied for both neural and symbolic parameter learning.

Energy Loss: A variety of differentiable loss functions can be chosen for $\mathcal{L}$. For simplicity, in this work, we present the *energy loss*. The energy loss parameter learning scheme directly minimizes the energy of the training samples, i.e., the per-sample losses are:

$$L_i(\mathbf{y}_i, \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl}) = E((\mathbf{y}_{i,t}, \mathbf{z}_i^*), \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl})$$

Notice that inference over the latent variables is necessary for gradient and objective value computations. However, a complete prediction from NeuPSL, i.e., inference over all components of $\mathbf{y}$, is unnecessary. Therefore the parameter learning problem is as follows:

$$\arg\min_{\mathbf{w}_{nn}, \mathbf{w}_{psl}} \sum_{i=1}^{P} \min_{\mathbf{z} \in \mathbf{\Omega}} \mathbf{w}_{psl}^T \Phi((\mathbf{y}_{i,t}, \mathbf{z}), \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn})$$

With L2 regularization, the NeuPSL energy function is strongly convex in all components of $\mathbf{y}_i$. Thus, by Danskin (1966), the gradient of the energy loss, $L_i(\cdot)$, with respect to $\mathbf{w}_{psl}$ at $\mathbf{y}_i, \mathbf{x}_i, \mathbf{x}_{nn,i}\mathbf{w}_{nn}$ is:

$$\nabla_{\mathbf{w}_{psl}} L_i(\mathbf{y}_i, \mathbf{x}_{sy,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl}) = \Phi((\mathbf{y}_{i,t}, \mathbf{z}_i^*), \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn})$$

Then the per-sample energy loss partial derivative with respect to $\mathbf{w}_{nn}[j]$ at $\mathbf{y}_i, \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{psl}$ is:

$$\frac{\partial L_i(\mathbf{y}_i, \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl})}{\partial \mathbf{w}_{nn}[j]} = \sum_{r=1}^{R} \mathbf{w}_{psl}[r] \sum_{q \in \tau_r} \frac{\partial \phi_q((\mathbf{y}_{i,t}, \mathbf{z}_i^*), \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn})}{\partial \mathbf{w}_{nn}[j]}$$

[1]Note arguments of the energy function and symbolic potentials are dropped for simplicity, i.e., $E(\cdot) = E(\mathbf{y}, \mathbf{x}_{sy,i}, \mathbf{x}_{nn,i}, \mathbf{w}_{nn}, \mathbf{w}_{psl})$, $\phi(\cdot) = \phi(\mathbf{y}, \mathbf{x}_{sy}, \mathbf{x}_{nn}, \mathbf{w}_{nn})$, and $\mathbf{g}_{nn}[i](\cdot) = \mathbf{g}_{nn}[i](\mathbf{x}_{nn}, \mathbf{w}_{nn})$.

Details on the learning algorithms and accounting for degenerate solutions of the energy loss are included in supplementary materials.

6 Experimental Evaluation

We evaluate NeuPSL's prediction performance and inference time on three tasks to demonstrate the significance of joint symbolic inference and learning. NeuPSL, implemented using the open-source PSL software package, can be integrated with any neural network library (here, we used TensorFlow).[2] Our investigation addresses the following questions: Q1) Can neuro-symbolic methods provide a boost over conventional purely data-driven neural models? Q2) Can we effectively leverage structural relationships across training examples through joint reasoning? Q3) How does NeuPSL compare with other neuro-symbolic methods in terms of time efficiency on large scale problems?

6.1 MNIST Addition

The first set of experiments are conducted on a variation of MNIST Addition, a widely used NeSy evaluation task [Manhaeve *et al.*, 2018]. The task involves determining the sum of two lists of MNIST images. For example, a **MNIST-Add1** addition is ($[\,\mathbf{3}\,] + [\,\mathbf{5}\,] = \mathbf{8}$), and a **MNIST-Add2** addition is ($[\,\mathbf{0}, \mathbf{4}\,] + [\,\mathbf{3}, \mathbf{7}\,] = \mathbf{41}$). The challenge stems from the lack of labels for the MNIST images in the addition equation. Only the final sum of the equation is given, leaving the task of identifying the individual digits and determining their values up to the model being used.

While NeuPSL proves to be successful in the original **MNIST-Add** setting (appendix for further details), here we are interested in exploring the power of joint inference and learning capabilities in NeSy systems. We introduce a variant of the **MNIST-Add** task in which digits are reused across multiple addition examples, i.e., we introduce *overlap*. Figure 2 demonstrates the process of introducing overlap and how joint models narrow the space of possible labels when MNIST images are re-used. For instance, in the scenario presented in Figure 2, the same MNIST image of a zero is utilized in two separate additions. To comply with both addition constraints, the potential label space is restricted and can no longer include options such as two or three, as they would violate one of the addition rules. In contrast, a model performing independent reasoning would have no way of enforcing this constraint across examples.

In the overlap variant of **MNIST-Add**, we focus on low-data settings to understand whether NeSy systems' joint reasoning can effectively leverage additional structure to overcome a lack of data. To create overlap, we begin with a set of n unique MNIST images from which we re-sample to create $(n+m)/2$ **MNIST-Add1** and $(n+m)/4$ **MNIST-Add2** additions. We vary the amount of overlap with $m \in \{0, n/2, n\}$ and compare performance with $n \in \{40, 60, 80\}$. Results

[2]Implementation details, hyperparameters, network architectures, hardware, and NeuPSL models, are described in the Appendix.

Code and Data: https://github.com/linqs/neupsl-ijcai23

Appendix: https://arxiv.org/abs/2205.14268

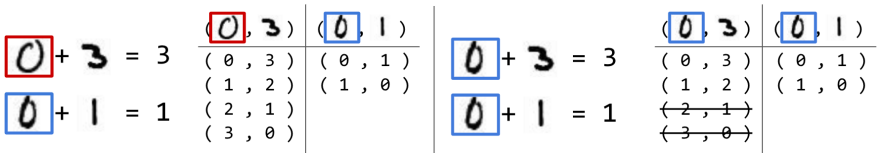

Figure 2: Example of overlapping MNIST images in **MNIST-Add1**. On the left, distinct images are used for each zero. On the right, the same image is used for both zeros.

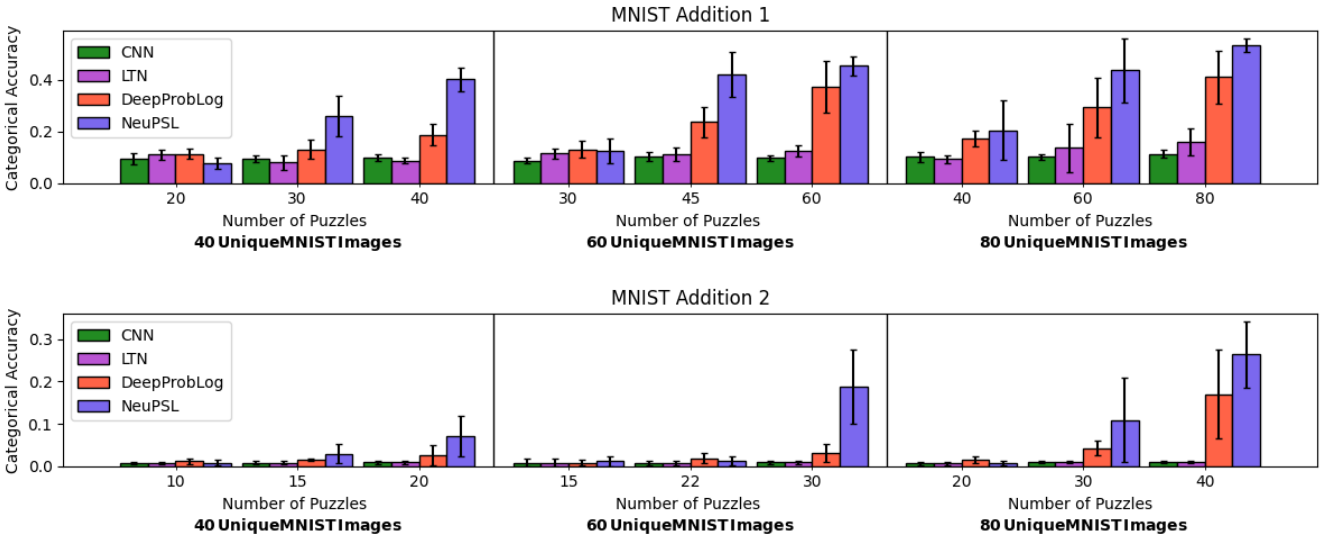

Figure 3: Average test set accuracy and standard deviation on **MNIST-Add** datasets with varying amounts of overlap.

are reported over ten test sets of $1,000$ MNIST images with overlap proportional to the respective train set.

Figure 3 summarizes average performance for varying overlap settings. Each panel varies the number of additions for a set number of unique MNIST images. For example, the upper left panel presents the results obtained for MNIST-Add1 with 40 unique images used to generate 20, 30, and 40 additions. Initially, there is not enough structure from the additions with no overlap for symbolic inference to discern the correct digit labels for training the neural models. Then, despite the number of unique MNIST images remaining the same, as the number of additions increases, DPL and NeuPSL improve their prediction performance by leveraging the added joint information (Q2). In all cases, NeuPSL performs best and uses the added structure most efficiently. LTNs and the CNN baseline benefit the least from joint information, a consequence of both learning and inference being performed independently across batches of additions (Q1).

6.2 Visual Sudoku Classification

Inspired by the Visual Sudoku problem proposed by Wang *et al.* (2019), Augustine *et al.* (2022) introduced a novel NeSy task, **Visual-Sudoku-Classification**. In this task, 4x4 Sudoku puzzles are constructed using unlabeled MNIST images. The model must identify whether a puzzle is correct, i.e., no duplicate digits in any row, column, or square. Therefore this task does not require learning the underlying label for images but rather whether an entire puzzle is valid. For instance, $\boxed{3}$ does not need to belong to a "3" class, instead $\boxed{3}$ and $\boxed{4}$ need to be labeled as different symbols. Similar to MNIST-Add we explore an overlap variant in low-data settings, with overlapping MNIST images across puzzles.

We compare NeuPSL with two baselines, CNN-Visual and CNN-Digit. The first, CNN-Visual, takes the pixels for a Sudoku puzzle as input and outputs the probability the puzzle is valid. The second, CNN-Digit, is provided the (unfair) advantage of all sixteen image labels as input. We use this to verify whether a neural model can learn Sudoku rules. Scalably developing LTN and DPL models in this new setting is not straightforward due to the large dimensionality of the output space. A non-expert implementation of a visual sudoku model in DPL and LTN may result in suboptimal reports on model performance and are therefore not included.

Figure 4 shows the accuracy of NeuPSL and CNN models on **Visual-Sudoku-Classification** with varying amounts of overlap. CNN-Visual and CNN-Digit struggle to leverage the problem structure and fail to generalize even the highest data and overlap setting with 256 MNIST images across 64 puzzles. However, NeuPSL achieves 70% accuracy using roughly 64 MNIST images across 16 puzzles, again showing it efficiently leverages joint information across training examples (Q1 and Q2). This is a particularly impressive result as the neural network in the NeuPSL model was trained to be a 93% 4-digit distinguisher without digit labels.

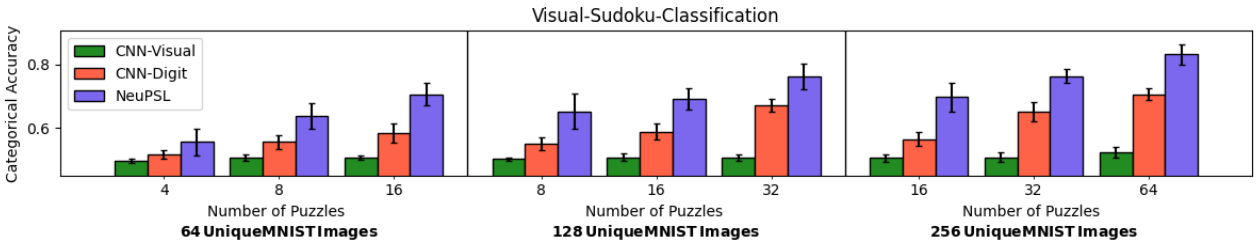

Figure 4: Average test set accuracy and standard deviation on **Visual-Sudoku-Classification** with varying amounts of overlap.

Method	Citeseer		Cora	
	(Accuracy)	(Seconds)	(Accuracy)	(Seconds)
Neural$_{PSL}$	57.76 ± 1.71	-	57.12 ± 2.13	-
LP$_{PSL}$	50.88 ± 1.18	-	73.32 ± 2.39	-
DeepProbLog	timeout	timeout	timeout	timeout
DeepStochLog	61.30 ± 1.44	34.42 ± 0.87	69.96 ± 1.47	165.28 ± 4.49
GCN	67.50 ± 0.57	3.10 ± 0.04	79.52 ± 1.13	1.31 ± 0.01
NeuPSL$_{LP}$	67.34 ± 1.17	3.98 ± 0.05	76.80 ± 2.27	4.00 ± 0.31
NeuPSL$_{LP+FP}$	**68.48 ± 1.22**	4.23 ± 0.05	**81.22 ± 0.79**	4.07 ± 0.14

Table 1: Test set accuracy and inference runtime in seconds on two citation network datasets.

6.3 Citation Network Node Classification

In our final experiment, we evaluate the performance of NeuPSL on two widely studied citation network node classification datasets: Citeseer and Cora [Sen *et al.*, 2008]. In these datasets, symbolic models have the potential to improve predictions by leveraging the homophilic structure of the citation network, i.e., two papers connected in the network are more likely to have the same label. This setting differs from **Visual-Sudoku-Classification** and **MNIST-Add** as the symbolic relations are not always true. Moreover, the symbolic relations can be defined over a general and potentially large number of nodes in the network, i.e., a node can be connected to any number of neighbors.

We propose two NeuPSL models for citation network node classification. Both models integrate a neural network that uses a paper's features to provide an initial classification, which is then adjusted via symbolic reasoning. The first model, NeuPSL$_{LP}$ (Label Propagation), directly uses the bag-of-words feature vector, while the second model, NeuPSL$_{LP+FP}$ (Label + Feature Propagation), first performs the feature construction procedure as described in Wu *et al.* (2019) to obtain a richer representation to provide to the neural model. We examine the runtime and model performance of NeSy methods NeuPSL$_{LP}$, NeuPSL$_{LP+FP}$, DPL and its scalable extension, DeepStochLog [Winters *et al.*, 2022], and a Graph Convolutional Network (GCN) [Kipf and Welling, 2017]. Additionally, we include the performance of two baselines, LP$_{PSL}$ and Neural$_{PSL}$. These baselines represent the distinct symbolic and neural components used in the NeuPSL$_{LP}$ model but perform only neural or symbolic reasoning, not both. We averaged the results over ten randomly sampled splits using 5% of the nodes for training, 5% of the nodes for validation, and 1000 nodes for testing.

Table 1 shows DeepStochLog, GCN, and NeuPSL all outperform the independent baselines (Q1), with NeuPSL$_{LP+FP}$ performing the best. These results demonstrate the power of using NeSy systems to effectively leverage structure to improve prediction performance. Additionally, NeuPSL is capable of scaling its joint inference process to larger structures, achieving higher accuracy with an 8 and 40 times speed up over DeepStochLog in Citeseer and Cora, respectively (Q3). Surprisingly, NeuPSL also achieves a higher prediction performance than even a GCN model while using significantly fewer trainable parameters.

7 Conclusion

In this paper, we introduced NeuPSL, a novel NeSy framework that integrates neural architectures and a tractable class of graphical models for jointly reasoning over symbolic relations and showed its utility across a range of neuro-symbolic tasks. There are many avenues for future work, including exploring different learning objectives, such as ones that balance traditional neural and energy-based losses and new application domains. Each of these is likely to provide new challenges and insights.

Acknowledgments

This work was partially supported by the National Science Foundation grant CCF-2023495 and a Google Faculty Research Award.

Contribution Statement

Connor Pryor and Charles Dickens contributed equally to this work.

References

[Alshukaili *et al.*, 2016] Duhai Alshukaili, Alvarao Fernandees, and Norman Paton. Structuring linked data search results using probabilistic soft logic. In *ISWC*, 2016.

[Augustine *et al.*, 2022] Eriq Augustine, Connor Pryor, Charles Dickens, Jay Pujara, William Yang Wang, and Lise Getoor. Visual sudoku puzzle classification: A suite of collective neuro-symbolic tasks. In *International Workshop on Neural-Symbolic Learning and Reasoning (NeSy)*, 2022.

[Bach *et al.*, 2017] Stephen Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss Markov random fields and probabilistic soft logic. *JMLR*, 18(1):1–67, 2017.

[Bader and Hitzler, 2005] Sebastian Bader and Pascal Hitzler. Dimensions of neural-symbolic integration - A structured survey. *arXiv preprint cs/0511042*, 2005.

[Badreddine *et al.*, 2022] Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *AI*, 303(4):103649, 2022.

[Beltagy *et al.*, 2014] Islam Beltagy, Katrin Erk, and Raymond Mooney. Probabilistic soft logic for semantic textual similarity. In *ACL*, 2014.

[Besold *et al.*, 2017] Tarek R. Besold, Artur S. d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro M. Domingos, Pascal Hitzler, Kai-Uwe Kühnberger, Luís C. Lamb, Daniel Lowd, Priscila Machado Vieira Lima, Leo de Penning, Gadi Pinkas, Hoifung Poon, and Gerson Zaverucha. Neural-symbolic learning and reasoning: A survey and interpretation. *arXiv preprint arXiv:1711.03902*, 2017.

[Bošnjak *et al.*, 2017] Matko Bošnjak, Tim Rocktäschel, Jason Naradowsky, and Sebastian Riedel. Programming with a differentiable forth interpreter. In *ICML*, 2017.

[Boyd *et al.*, 2010] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[Cohen *et al.*, 2020] William W. Cohen, Fan Yang, and Kathryn Mazaitis. Tensorlog: A probabilistic database implemented using deep-learning infrastructure. *JAIR*, 67:285–325, 2020.

[Danskin, 1966] John Danskin. The theory of max-min, with applications. *SIAM Journal on Applied Mathematics*, 14(4):641–664, 1966.

[d'Avila Garcez *et al.*, 2002] Artur S. d'Avila Garcez, Krysia Broda, and Dov M. Gabbay. *Neural-Symbolic Learning Systems: Foundations and Applications*. Springer, 2002.

[d'Avila Garcez *et al.*, 2009] Artur S. d'Avila Garcez, Luís C. Lamb, and Dov M. Gabbay. *Neural-Symbolic Cognitive Reasoning*. Springer, 2009.

[d'Avila Garcez *et al.*, 2019] Artur d'Avila Garcez, Marco Gori, Luís C. Lamb, Luciano Serafini, Michael Spranger, and Son N. Tran. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *Journal of Applied Logics*, 6(4):611–632, 2019.

[De Raedt *et al.*, 2007] Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. Problog: A probabilistic prolog and its application in link discovery. In *IJCAI*, 2007.

[De Raedt *et al.*, 2020] Luc De Raedt, Sebastijan Dumančić, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. In *IJCAI*, 2020.

[Deng and Wiebe, 2015] Lingjia Deng and Janyce Wiebe. Joint prediction for Entity/Event-LEvel sentiment analysis using probabilistic soft logic models. In *EMNLP*, 2015.

[Dickens *et al.*, 2020] Charles Dickens, Rishika Singh, and Lise Getoor. Hyperfair: A soft approach to integrating fairness criteria. In *FAccTRec*, 2020.

[Diligenti *et al.*, 2017] Michelangelo Diligenti, Soumali Roychowdhury, and Marco Gori. Integrating prior knowledge into deep learning. In *ICMLA*, 2017.

[Donadello *et al.*, 2017] Ivan Donadello, Luciano Serafini, and Artur S. d'Avila Garcez. Logic tensor networks for semantic image interpretation. In *IJCAI*, 2017.

[Evans and Grefenstette, 2018] Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *JAIR*, 61:1–64, 2018.

[Farnadi *et al.*, 2019] Golnoosh Farnadi, Behrouz Babaki, and Lise Getoor. A declarative approach to fairness in relational domains. *IEEE Data Engineering Bulletin*, 42(3):36–48, 2019.

[Hu *et al.*, 2016] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing deep neural networks with logic rules. In *ACL*, 2016.

[Kimmig *et al.*, 2019] Angelika Kimmig, Alex Memory, Renée J. Miller, and Lise Getoor. A collective, probabilistic approach to schema mapping using diverse noisy evidence. *TKDE*, 31(8):1426–1439, 2019.

[Kipf and Welling, 2017] Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[Klir and Yuan, 1995] George J. Klir and Bo Yuan. *Fuzzy Sets and Fuzzy Logic - Theory and Applications*. Prentice Hall, 1995.

[Kouki *et al.*, 2015] Pigi Kouki, Shobeir Fakhraei, James R. Foulds, Magdalini Eirinaki, and Lise Getoor. Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In *RecSys*, 2015.

[Lamb *et al.*, 2020] Luís C. Lamb, Artur d'Avila Garcez, Marco Gori, Marcelo O. R. Prates, Pedro H. C. Avelar, and Moshe Y. Vardi. Graph neural networks meet neural-symbolic computing: A survey and perspective. In *IJCAI*, 2020.

[LeCun *et al.*, 2006] Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu Jie Huang. A tutorial on energy-based learning. *Predicting Structured Data*, 1(0), 2006.

[Liang *et al.*, 2017] Chen Liang, Jonathan Berant, Quoc Le, Kenneth Forbus, and Ni Lao. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In *ACL*, 2017.

[Liu *et al.*, 2016] Shulin Liu, Kang Liu, Shizhu He, and Jun Zhao. A probabilistic soft logic based approach to exploiting latent and global information in event classification. In *AAAI*, 2016.

[Manhaeve *et al.*, 2018] Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. DeepProbLog: Neural probabilistic logic programming. In *NeurIPS*, 2018.

[Manhaeve *et al.*, 2021] Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Neural probabilistic logic programming in DeepProbLog. *AI*, 298:103504, 2021.

[Mao *et al.*, 2019] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *ICLR*, 2019.

[Mehta *et al.*, 2018] Sanket Vaibhav Mehta, Jay Yoon Lee, and Jaime Carbonell. Towards semi-supervised learning for deep semantic role labeling. In *EMNLP*, 2018.

[Nandwani *et al.*, 2019] Yatin Nandwani, Abhishek Pathak, and Parag Singla. A primal dual formulation for deep learning with constraints. In *NeurIPS*, 2019.

[Pujara *et al.*, 2013] Jay Pujara, Hui Miao, Lise Getoor, and William W. Cohen. Knowledge graph identification. In *ISWC*, 2013.

[Rocktäschel and Riedel, 2017] Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. In *NeurIPS*, 2017.

[Rospocher, 2018] Marco Rospocher. An ontology-driven probabilistic soft logic approach to improve nlp entity annotation. In *ISWC*, 2018.

[Sachan *et al.*, 2018] Mrinmaya Sachan, Kumar Avinava Dubey, Tom M Mitchell, Dan Roth, and Eric P Xing. Learning pipelines with limited data and domain knowledge: A study in parsing physics problems. In *NeurIPS*, 2018.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Mark Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[Serafini and d'Avila Garcez, 2016] Luciano Serafini and Artur S. d'Avila Garcez. Learning and reasoning with logic tensor networks. In *AI*IA*, 2016.

[Sikka *et al.*, 2020] Karan Sikka, Andrew Silberfarb, John Byrnes, Indranil Sur, Ed Chow, Ajay Divakaran, and Richard Rohwer. Deep adaptive semantic logic (dasl): Compiling declarative knowledge into deep neural networks. Technical report, SRI International, 2020.

[Sridhar *et al.*, 2018] Dhanya Sridhar, Jay Pujara, and Lise Getoor. Scalable probabilistic causal structure discovery. In *IJCAI*, 2018.

[Wang *et al.*, 2019] Po-Wei Wang, Priya Donti, Bryan Wilder, and Zico Kolter. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *ICML*, 2019.

[Winters *et al.*, 2022] Thomas Winters, Giuseppe Marra, Robin Manhaeve, and Luc De Raedt. DeepStochLog: Neural stochastic logic programming. In *AAAI*, 2022.

[Wu *et al.*, 2019] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *ICML*, 2019.

[Xu *et al.*, 2018] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. A semantic loss function for deep learning with symbolic knowledge. In *ICML*, 2018.

[Yang *et al.*, 2017] Fan Yang, Zhilin Yang, and William W. Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *NeurIPS*, 2017.

[Yang *et al.*, 2020] Zhun Yang, Adam Ishay, and Joohyung Lee. Neurasp: Embracing neural networks into answer set programming. In *IJCAI*, 2020.