

# Improving neural classification with Logical Prior Knowledge

Arthur Ledaguenel<sup>a, b, \*</sup>, Celine Hudelot<sup>b</sup> and Mostepha Khouadjia<sup>a</sup>

<sup>a</sup>IRT SystemX

<sup>b</sup>MICS, CentraleSupélec

**Abstract.** Neurosymbolic AI is a growing field of research aiming to combine neural networks learning capabilities with the reasoning abilities of symbolic systems. In this paper, we propose a new formalism for supervised multi-label classification informed by propositional prior knowledge. We introduce a new neurosymbolic technique called semantic conditioning at inference, which only constrains the system during inference while leaving the training unaffected. We discuss its theoretical and practical advantages over two other popular neurosymbolic techniques: semantic conditioning and semantic regularization. We develop a new multi-scale methodology to evaluate how the benefits of a neurosymbolic technique evolve with the scale of the network. We then evaluate experimentally and compare the benefits of all three techniques across model scales on several datasets. Our results demonstrate that semantic conditioning at inference can leverage prior knowledge to build more accurate neural-based systems compared to an uninformed system. We show that despite only working at inference, it retains a substantial portion of the benefits offered by semantic conditioning. Furthermore, we detail several use cases in which semantic conditioning at inference can be applied while semantic conditioning cannot.

## 1 Introduction

Neurosymbolic AI is a growing field of research aiming to combine neural network learning capabilities with the reasoning abilities of symbolic systems. This hybridization can take many shapes depending on how the neural and symbolic components interact, like shown in [22, 41].

An important sub-field of neurosymbolic AI is Informed Machine Learning [40], which studies how to leverage prior knowledge to improve neural-based systems. There again, proposed techniques in the literature can be of very different nature depending on the type of task (*e.g.* regression, classification, detection, generation, etc.), the formalism used to represent the prior knowledge (*e.g.* mathematical equations, knowledge graphs, logical formulas, etc.), the stage at which knowledge is embedded (*e.g.* data processing, neural architecture design, learning procedure, inference procedure, etc.) and benefits expected from the hybridization (*e.g.* explainability, performance, frugality, etc.).

In this paper, we tackle **supervised** multi-label **classification** tasks where the set of semantically valid outputs is specified by a **propositional formula**. We consider neurosymbolic techniques that integrate prior knowledge during learning, inference or both and that

mainly aim at improving the performance of a neural classification system. More specifically, we study a family of neurosymbolic techniques that leverage probabilistic reasoning to integrate prior knowledge, a trend that has gained significant traction in the recent literature [42, 27, 1, 2].

**Contributions** First, we introduce a formalism for supervised classification informed by propositional prior knowledge. Then, we build upon this formalism to re-frame two existing neurosymbolic techniques: one that only impacts training (semantic regularization) and one that impacts both training and inference (semantic conditioning). We also define a new technique that only impacts the inference stage: **semantic conditioning at inference**. To the best of our knowledge, we are the first to define a neurosymbolic technique based on probabilistic reasoning which only impacts inference. This is a particularly useful property in the era of **off-the-shelves** and **foundation** models [7], which are pre-trained on massive amounts of general data to then be applied in a multitude of heterogeneous downstream tasks. We also show that this makes semantic conditioning at inference more tractable than the two other techniques. Finally, we develop a multi-scale evaluation methodology which enables to study how the benefits of neurosymbolic techniques evolve with the scale of the neural network and evaluate all techniques on four datasets, including two large datasets whose size is rarely encountered in the neurosymbolic literature. We show experimentally that benefits from semantic conditioning at inference do not vanish as the neural network scales.

**Outline** We start with preliminary notions in Section 2 and related works in Section 3. Then, in Section 4, we introduce our formalism for representing supervised informed classification tasks (Section 4.1) and neurosymbolic techniques (Section 4.2), then compare their theoretical properties (Section 4.3), implementation and complexity (Section 4.4). In Section 5, we present our multi-scale evaluation methodology and analyze the results for all techniques on four different tasks. We conclude with possible future research questions in Section 6. Proofs for all stated propositions can be found in the supplementary materials.

## 2 Preliminaries

In this section, we give a formal definition of a neural classification system, which serves as our basis for neurosymbolic techniques. Then we briefly introduce propositional logic, one of the most common language for representing knowledge about a set of binary variables, which we will use throughout the paper to express our prior

---

\* Corresponding Author. Email: arthur.ledaguenel@irt-systemx.fr

knowledge on multi-label classification tasks. Finally, we define several probabilistic reasoning problems that will be the foundation of the neurosymbolic techniques we will study in the paper.

## 2.1 Neural classification

In supervised machine learning, the objective is to learn a functional relationship  $f : \mathcal{X} \mapsto \mathcal{Y}$  between an **input domain**  $\mathcal{X}$  and an **output domain**  $\mathcal{Y}$  from a labeled dataset  $D := (x^i, \mathbf{y}^i)_{1 \leq i \leq d} \in (\mathcal{X} \times \mathcal{Y})^d$ . Deep learning systems usually tackle such tasks using two main modules: a neural network (*i.e.*, a parametric and differentiable computational graph)  $M$  is designed based on assumed properties of  $f$ , then a differentiable cost function  $L$  is used to measure the distance between the predictions and the labels, and the weights of the network are optimized using backpropagation and gradient descent to minimize the empirical error on the training set.

However, for classification tasks, such a framework cannot be applied strictly. Multi-label classification is a type of learning tasks where elements in the output domain  $\mathcal{Y}$  are subsets of a finite set of classes  $\mathbf{Y}$ . We call such a subset a **state**, and note  $\mathcal{Y} = \mathbb{B}^{\mathbf{Y}}$ , where  $\mathbb{B} = \{0, 1\}$ . A state  $\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}$  can also be seen as an application that maps each variables to  $\mathbb{B}$  (*i.e.*, for a variable  $V \in \mathbf{Y}$ , we note  $\mathbf{y}(V) = 1$  is equivalent to  $V \in \mathbf{y}$ ). Since the output space is discrete, a differentiable distance cannot be defined directly on it.

Hence, a slightly modified framework is adopted, where a third module  $l$  (besides the model and loss modules), called the inference module, has to be defined to bridge the gap between the continuous nature of the neural network (needed for gradient descent) and the discrete nature of the output space. This third module, although essential, is not often explicitly described. An illustration can be found on Figure 1.

**Definition 1.** A **neural classification system** for multi-label classification is the given of :

- a **parametric differentiable** (*i.e.*, neural) module  $M$ , called the **model**, which takes as inputs  $x \in \mathbb{R}^d$ , parameters  $\theta \in \Theta$  and outputs  $M(x, \theta) := M_\theta(x) := \mathbf{a} \in \mathbb{R}^k$ , called **pre-activation scores** or **logits**.
- a **non-parametric differentiable** module  $L$ , called the **loss** module, which takes  $\mathbf{a} \in \mathbb{R}^k$  and  $\mathbf{y} \in \{0, 1\}^k$  as inputs and outputs a scalar.
- a **non-parametric** module  $l$ , called the **inference** module, which takes  $\mathbf{a} \in \mathbb{R}^k$  as input and outputs a prediction  $\hat{\mathbf{y}} \in \{0, 1\}^k$ .

*Remark 1.* For lighter notations, we note  $\mathbf{a} \in \mathbb{R}^k$  as a simpler notation for  $\mathbf{a} \in \mathbb{R}^{\mathbf{Y}}$  assuming  $\mathbf{Y} := \{Y_j\}_{1 \leq j \leq k}$ .

A common approach to design a neural classification system is to build upon a natural probabilistic interpretation. Logits produced by the neural network are seen as parameters of a conditional probability distribution of the output given the input  $\mathcal{P}(\cdot | M_\theta(x))$ , the loss module computes the cross-entropy of that distribution with a ground truth label, and the inference module computes the most probable output given the learned distribution.

When no prior knowledge is available about the set of classes (un-informed case), a standard hypothesis is to assume independent output variables. We illustrate below how this translates in a specific neural classification system.

**Example 1.** For **independent multi-label classification**, we apply a **sigmoid layer on logits to turn them into probability scores**. The loss is the **binary cross-entropy (BCE)** between probability scores and

labels, and a variable is predicted to be true if its probability is above 0.5 (or equivalently its logit is above 0). This results in the following modules:

$$L_{imc}(\mathbf{a}, \mathbf{y}) := \text{BCE}(\mathbf{s}(\mathbf{a}), \mathbf{y}) = - \sum_j y_j \cdot \log(\mathbf{s}(a_j)) + (1 - y_j) \cdot \log(1 - \mathbf{s}(a_j)) \quad (1)$$

$$l_{imc}(\mathbf{a}) := \mathbb{1}[\mathbf{a} \geq 0] \quad (2)$$

where  $\mathbf{s}(a_i) = \frac{e^{a_i}}{1+e^{a_i}}$  is the **sigmoid function** and  $\mathbb{1}[z] := \begin{cases} 1 & \text{if } z \text{ true} \\ 0 & \text{otherwise} \end{cases}$  the **indicator function**.

## 2.2 Propositional Logic

A **propositional signature** is a set  $\mathbf{Y}$  of symbols called **variables** (*e.g.*  $\mathbf{Y} = \{a, b\}$ ). A **propositional formula** is formed inductively from variables and other formulas by using unary ( $\neg$ , which expresses negation) or binary ( $\vee, \wedge$ , which express disjunction and conjunction respectively) connectives (*e.g.*  $\kappa = a \wedge b$  which is **true** if both variables  $a$  and  $b$  are **true**). We note  $\mathcal{F}(\mathbf{Y})$  the set of formulas that can be formed in this way. A state  $\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}$  can be inductively extended to define a **valuation**  $\mathbf{y}^*$  on all formulas using the standard semantics of propositional logic (*e.g.*  $\mathbf{y}^*(a \wedge b) = \mathbf{y}(a) \times \mathbf{y}(b)$ ). We say that a state  $\mathbf{y}$  **satisfies** a formula  $\kappa$ , noted  $\mathbf{y} \models \kappa$ , if  $\mathbf{y}^*(\kappa) = 1$ . We say that a formula is **satisfiable** when it is satisfied by at least one state. We use the symbol  $\top$  to represent **tautologies** (*i.e.*, formulas which are satisfied by all states). Two formulas  $\kappa$  and  $\gamma$  are said **equivalent**, noted  $\kappa \equiv \gamma$ , if they are satisfied by exactly the same states. We refer to [36] for more details on propositional logic.

## 2.3 Probabilistic reasoning

One challenge of neurosymbolic AI is to bridge the gap between the discrete nature of logic and the continuous nature of neural networks. Probabilistic reasoning can provide the interface between these two realms by allowing us to reason about uncertain facts. In this section, we introduce two probabilistic reasoning problems: **Probabilistic Query Estimation (PQE)**, *i.e.*, computing the probability of a formula to be satisfied, and **Most Probable Explanation (MPE)**, *i.e.*, finding the most probable state that satisfies a given formula.

A probability distribution on a set of **boolean variables**  $\mathbf{Y}$  is an application  $\mathcal{P} : \mathbb{B}^{\mathbf{Y}} \mapsto \mathbb{R}^+$  that maps each state  $\mathbf{y}$  to a probability  $\mathcal{P}(\mathbf{y})$  such that  $\sum_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{P}(\mathbf{y}) = 1$ . To define internal operations between distributions, like multiplication, we extend this definition to un-normalized distributions  $\mathcal{E} : \mathbb{B}^{\mathbf{Y}} \mapsto \mathbb{R}^+$ . The **null distribution** is the application that maps all states to 0. The **partition function**  $Z : \mathcal{E} \mapsto \sum_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{E}(\mathbf{y})$  maps each distribution to its sum, and we note  $\bar{\mathcal{E}} := \frac{\mathcal{E}}{Z(\mathcal{E})}$  the normalized distribution (when  $\mathcal{E}$  is non-null). The **mode** of a distribution  $\mathcal{E}$  is its most probable state, ie  $\text{argmax}_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{E}(\mathbf{y})$ .

The independent multi-label classification system (see Example 1) is build by following the probabilistic interpretation based on the **exponential probability distribution**, which is parameterized by a vector of logits  $\mathbf{a} \in \mathbb{R}^k$ , one for each variable in  $\mathbf{Y}$ , and corresponds to the joint distribution of independent Bernoulli variables  $\mathcal{B}(p_i)_{1 \leq i \leq k}$  with  $p_i = \mathbf{s}(a_i)$ .

**Definition 2.** Given a vector  $\mathbf{a} \in \mathbb{R}^k$ , the **exponential distribution** is:

$$\mathcal{E}(\cdot | \mathbf{a}) : \mathbf{y} \mapsto \prod_{1 \leq i \leq k} e^{a_i \cdot y_i} \quad (3)$$

We will note  $\mathcal{P}(\cdot|\mathbf{a}) = \overline{\mathcal{E}(\cdot|\mathbf{a})}$  the corresponding normalized probability distribution.

Typically, when belief about random variables is expressed through a probability distribution and new information is collected in the form of evidence (*i.e.*, a partial assignment of the variables), we are interested in two things: computing the probability of such evidence and updating our beliefs using Bayes' rules by conditioning the distribution on the evidence. Probabilistic reasoning allows us to perform the same operations with logical knowledge in place of evidence. Let's assume a probability distribution  $\mathcal{P}$  on variables  $\mathbf{Y} := \{Y_j\}_{1 \leq j \leq k}$  and a **satisfiable** propositional formula  $\kappa$ . Notice that  $\mathcal{P}$  defines a probability distribution on the set of states of  $\mathbf{Y}$ . We also note  $\mathbb{1}_\kappa$  the indicator function of  $\kappa$  which maps satisfying states to 1 and others to 0:

$$\mathbb{1}_\kappa(\mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{y} \models \kappa \\ 0 & \text{otherwise} \end{cases}$$

**Definition 3.** The **probability** of  $\kappa$  under  $\mathcal{P}$  is:

$$\mathcal{P}(\kappa) := Z(\mathcal{P} \cdot \mathbb{1}_\kappa) = \sum_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{P}(\mathbf{y}) \cdot \mathbb{1}_\kappa(\mathbf{y}) \quad (4)$$

The distribution  $\mathcal{P}$  **conditioned on**  $\kappa$ , noted  $\mathcal{P}(\cdot|\kappa)$ , is:

$$\mathcal{P}(\cdot|\kappa) := \overline{\mathcal{P} \cdot \mathbb{1}_\kappa} \quad (5)$$

Since  $\mathcal{P}(\cdot|\mathbf{a})$  is strictly positive (for all  $\mathbf{a}$ ), if  $\kappa$  is satisfiable then its probability under  $\mathcal{P}(\cdot|\mathbf{a})$  is also strictly positive. We note:

$$\begin{aligned} \mathcal{P}(\kappa|\mathbf{a}) &:= Z(\mathcal{P}(\cdot|\mathbf{a}) \cdot \mathbb{1}_\kappa) \\ \mathcal{P}(\cdot|\mathbf{a}, \kappa) &:= \frac{\mathcal{P}(\cdot|\mathbf{a}) \cdot \mathbb{1}_\kappa}{\mathcal{P}(\kappa|\mathbf{a})} \end{aligned}$$

Computing  $\mathcal{P}(\kappa|\mathbf{a})$  is a **counting** problem called **Probabilistic Query Estimation (PQE)**. Computing the mode of  $\mathcal{P}(\cdot|\mathbf{a}, \kappa)$  is an **optimization** problem called **Most Probable Explanation (MPE)**. Notice that computing  $\mathcal{P}(\mathbf{y}|\mathbf{a}, \kappa)$  for a satisfying state  $\mathbf{y} \models \kappa$  is equivalent to solving PQE because  $\mathcal{P}(\mathbf{y}|\mathbf{a})$  can be computed in polynomial time and:

$$\mathcal{P}(\mathbf{y}|\mathbf{a}, \kappa) = \frac{\mathcal{P}(\mathbf{y}|\mathbf{a})}{\mathcal{P}(T|\mathbf{a})}$$

Solving these probabilistic reasoning problems is at the core of many neurosymbolic techniques, as shown in Section 4.

### 3 Related work

In this paper, we restricted our formalism to supervised classification tasks informed with propositional prior knowledge. However, many techniques in the literature work with prior knowledge expressed in a different language or solve classification tasks where full supervision is lacking.

**Alternative logics** Besides propositional logic, there are many languages used in the neurosymbolic literature to represent logical prior knowledge: HEX-graphs in [12], tractable circuits in [1], linear programs in [31], Prolog in [27], ASP in [43], First Order Logic in [5]. Many translation methods exist to convert knowledge expressed in those alternative logics into propositional logic. The trade-off between representation languages is mainly between expressivity and tractability.

**Supervision settings** In real world applications, labeling large amounts of data is difficult, expensive and slow, especially for multi-label classification tasks featuring many classes [11]. Therefore, much work has been done to formalize and exploit cheaper supervision settings where input samples are not fully labeled. In the **semi-supervised** setting [37, 19], only a fraction of input samples are fully labeled while the rest is unlabeled. Closely related is the **partial-labels** setting [14], where only a subset of the classes are labeled for each input sample. Partial labels can typically be found when prior knowledge represents a functional dependency between a set of latent variables and a set of observed variables, like in the MNIST-Add task [27, 5, 26, 39], which aim is to learn a latent representation of handwritten digits from observing only their sum. Some neurosymbolic techniques have been specifically designed for these supervision settings [42, 2].

## 4 Unifying neurosymbolic techniques for informed supervised classification

### 4.1 Task

In this paper, we say that a task of supervised multi-label classification is **informed** when it comes attached with prior knowledge, expressed as a propositional formula  $\kappa$ , that specifies which states in the output space are **semantically valid**.

A supervised dataset  $D$  is **consistent** with prior knowledge  $\kappa$  if all labels satisfy  $\kappa$  (*i.e.*,  $\forall 1 \leq i \leq n, \mathbf{y}^i \models \kappa$ ). In this paper we will work under the hypothesis that both training and test datasets are consistent. However, some techniques allow for a relaxation of this assumption, enabling to use inconsistent datasets.

### 4.2 Techniques

This paper does not discuss the architecture of the neural model (*e.g.* fully connected, convolutional, transformer-based, etc.) which mainly depends on the modality of the input space (*e.g.* images, texts, etc.), but rather focuses on the two other modules, to embed the structural prior we have on the output space. We give below two examples of informed classification tasks and how the loss and inference modules can be adapted to embed prior knowledge.

**Example 2.** *Categorical classification arises when one and only one output variable is true for a given input sample (e.g. mapping an image to a single digit in  $\llbracket 0, 9 \rrbracket$  for MNIST). These constraints can easily be enforced by the following propositional formula:*

$$\kappa_{\odot_k} := \left( \bigvee_{1 \leq j \leq k} Y_j \right) \wedge \left( \bigwedge_{1 \leq j < l \leq k} (\neg Y_j \vee \neg Y_l) \right) \quad (6)$$

where the first part ensures that at least one variable is true and the second part prevents two variables to be true simultaneously. For categorical classification, the sigmoid layer is replaced by a softmax layer and the variable with the maximum score is predicted, which leads to the following modules:

$$L_{\odot_k}(\mathbf{a}, \mathbf{y}) := \text{CE}(\mathbf{s}(\mathbf{a}), \mathbf{y}) = -\log(\langle \sigma(\mathbf{a}), \odot_k(j) \rangle) \quad (7)$$

$$l_{\odot_k}(\mathbf{a}) := \odot_k(\text{argmax}(\mathbf{a})) \quad (8)$$

where  $\text{CE}$  is the cross-entropy,  $\sigma(\mathbf{a}) = \left( \frac{e^{a_j}}{\sum_l e^{a_l}} \right)_{1 \leq j \leq k}$  and  $\odot_k$  gives the one-hot encoding (starting at 1) of  $j \in \llbracket 1, k \rrbracket$ , *e.g.*  $\odot_4(2) = (0, 1, 0, 0)$ .

**Example 3. Hierarchical classification on a set of variables**  $\{Y_j\}_{1 \leq j \leq k}$  is usually formulated with a directed acyclic graph  $G = (Y, E_h)$  where the nodes are the variables and the edges  $E_h$  express subsumption between those variables (e.g. a dog is an animal). This formalism can even be enriched with exclusion edges  $H = (Y, E_h, E_e)$  (e.g. an input cannot be both a dog and a cat), like in HEX-graphs [12]. There again, the translation to propositional logic is straightforward:

$$\kappa_H := \left( \bigwedge_{(i,j) \in E_h} Y_i \vee \neg Y_j \right) \wedge \left( \bigwedge_{(i,j) \in E_e} (\neg Y_i \vee \neg Y_j) \right) \quad (9)$$

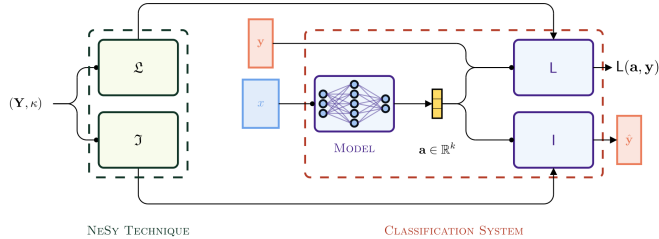
where the first part ensures that a son node cannot be true if its father node is not and the second part prevents two mutually exclusive nodes to be true simultaneously. Many techniques have been proposed to integrate hierarchical knowledge in a neural classification system. For instance, [30] introduces a hierarchical loss to penalize more errors on higher classes of the hierarchy, [17] refines the logits based on the hierarchy while [12] replaces the exponential distribution by a Conditional Random Field that integrates the hierarchical knowledge.

Beyond categorical and hierarchical classification, propositional logic can be used to define very diverse output spaces (e.g. Sudoku solutions [4], simple paths in a graph [42, 1], preference rankings [42], matchings in a graph [33, 2], etc.).

Therefore, the purpose of a neurosymbolic technique is to automatically derive appropriate loss and inference modules from prior knowledge, **generalizing the work made on independent, categorical and hierarchical cases to arbitrary structures**. We formalize this process with the definition below and illustrate it on Figure 1.

**Definition 4** (Supervised neurosymbolic technique). A supervised (model agnostic) **neurosymbolic technique** is  $\mathfrak{T} := (\mathcal{L}, \mathcal{J})$  such that for any finite set of variables  $\mathbf{Y}$  and prior knowledge  $\kappa \in \mathcal{F}(\mathbf{Y})$ :

$$\begin{aligned} \mathcal{L}(\mathbf{Y}, \kappa) &:= L : \mathbb{R}^k \times \mathcal{Y} \mapsto \mathbb{R}^+ \\ \mathcal{J}(\mathbf{Y}, \kappa) &:= I : \mathbb{R}^k \mapsto \mathcal{Y} \end{aligned}$$



**Figure 1.** Illustration of a (model agnostic) neurosymbolic technique: it takes prior knowledge as input and outputs the loss and inference modules of a neural classification system.

We define illustrate this definition below with three neurosymbolic techniques, including our new technique called **semantic conditioning at inference**, and relate each technique to the existing neurosymbolic literature.

**Semantic conditioning** Following the probabilistic interpretation introduced in Section 2.1, a natural way to integrate prior knowledge  $\kappa$  into a neural classification system is to condition the distribution  $\mathcal{P}(\cdot | M(x, \theta))$  on  $\kappa$ . This conditioning affects the loss and inference modules, both underpinned by the conditional distribution. It

was first introduced in [12] for Hierarchical-Exclusion (HEX) graphs constraints. Semantic probabilistic layers [1] can be used to implement semantic conditioning on tractable circuits. Moreover they go beyond exponential distributions and allow for a more expressive family of distributions using probabilistic circuits. NeurASP [43] defines semantic conditioning on a predicate extension of ASP programs. Likewise, DeepProbLog [27] provides an interface between Problog [10] programs and neural networks. However, since probabilistic reasoning in DeepProbLog is performed through grounding, its computational complexity is akin to that of semantic conditioning where the set of variables is the Herbrand base of the Prolog program. An approached method for semantic conditioning on linear programs is proposed in [31].

**Definition 5. Semantic conditioning** is  $\mathfrak{T}_{sc} := (\mathcal{L}_{sc}, \mathcal{J}_{sc})$  such that for any finite set of variables  $\mathbf{Y}$  and formula  $\kappa \in \mathcal{F}(\mathbf{Y})$ :

$$\mathcal{L}_{sc}(\mathbf{Y}, \kappa) : (\mathbf{a}, \mathbf{y}) \rightarrow -\log(\mathcal{P}(\mathbf{y} | \mathbf{a}, \kappa)) \quad (10)$$

$$\mathcal{J}_{sc}(\mathbf{Y}, \kappa) : \mathbf{a} \rightarrow \operatorname{argmax}_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{P}(\mathbf{y} | \mathbf{a}, \kappa) \quad (11)$$

**Semantic regularization** An other approach, and one of the most popular in the literature, is to use a multi-objective scheme to train the neural network on both labeled instances and semantic constraints: a regularization term measuring the *consistency* of the output of the neural network with the prior knowledge is added to the standard negative log-likelihood of the labels to steer the model towards producing scores that match the prior knowledge. First introduced using fuzzy logics [13, 16, 5], a regularization technique based on probabilistic reasoning was introduced in [42].

**Definition 6. Semantic regularization** (with coefficient  $\lambda > 0$ ) is  $\mathfrak{T}_r^\lambda := (\mathcal{L}_r^\lambda, \mathcal{J}_r^\lambda)$  such that for any finite set of variables  $\mathbf{Y}$  and formula  $\kappa \in \mathcal{F}(\mathbf{Y})$ :

$$\mathcal{L}_r^\lambda(\mathbf{Y}, \kappa) : (\mathbf{a}, \mathbf{y}) \rightarrow -\log(\mathcal{P}(\mathbf{y} | \mathbf{a})) - \lambda \cdot \log(\mathcal{P}(\kappa | \mathbf{a})) \quad (12)$$

$$\mathcal{J}_r^\lambda(\mathbf{Y}, \kappa) : \mathbf{a} \rightarrow \operatorname{argmax}_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{P}(\mathbf{y} | \mathbf{a}) \quad (13)$$

**Semantic conditioning at inference** Finally, we introduce a new neurosymbolic technique, called **semantic conditioning at inference**, which is derived from semantic conditioning but only applies conditioning in the inference module (*i.e.*, infers the most probable state that satisfies prior knowledge) while retaining the standard negative log-likelihood loss.

**Definition 7. Semantic conditioning at inference** is  $\mathfrak{T}_{sci} := (\mathcal{L}_{sci}, \mathcal{J}_{sci})$  such that for any finite set of variables  $\mathbf{Y}$  and formula  $\kappa \in \mathcal{F}(\mathbf{Y})$ :

$$\mathcal{L}_{sci}(\mathbf{Y}, \kappa) : (\mathbf{a}, \mathbf{y}) \rightarrow -\log(\mathcal{P}(\mathbf{y} | \mathbf{a})) \quad (14)$$

$$\mathcal{J}_{sci}(\mathbf{Y}, \kappa) = \mathcal{J}_{sc}(\mathbf{Y}, \kappa) \quad (15)$$

### 4.3 Properties

Our formalism enables us to define desirable properties of neurosymbolic techniques in a general way. We detail below two critical properties and identify which of the three techniques possess them. Formal definitions for the properties and proofs of the propositions can be found in the supplementary materials.

**Syntactic invariance** A neurosymbolic technique is **invariant to syntax** when equivalent formulas produce identical loss and inference modules when fed to  $\mathcal{L}$  and  $\mathcal{J}$ .

**Proposition 4.** *Semantic regularization, semantic conditioning and semantic conditioning at inference are all invariant to syntax.*

This is because all three techniques rely on probabilistic reasoning tasks which only depend on the indicator function  $\mathbb{1}_\kappa$  of the formula (two equivalent formulas have identical indicator functions by definition). This property is generally not satisfied by neurosymbolic techniques based on fuzzy logics [13, 16, 5].

**Consistency** A neurosymbolic technique is **consistent** (see [1]) when the inference module can only produce outputs that satisfy the prior knowledge.

**Proposition 5.** *Both semantic conditioning and semantic conditioning at inference are consistent.*

This is not true of semantic regularization which can not guarantee consistency with the independent inference module.

Furthermore, when performing inference based on identical model modules and learned parameters, semantic conditioning at inference **guarantees** greater or equal accuracy compared to traditional independent inference.

**Proposition 6.** *if  $\mathcal{I}_{imc}$  infers the right labels, then  $\mathcal{I}_\kappa$  will also infer the right labels, i.e., :*

$$\forall \mathbf{a} \in \mathbb{R}^k, \mathbf{y} \models \kappa, \mathcal{I}_{imc}(\mathbf{a}) = \mathbf{y} \implies \mathcal{I}_{sci}(\mathbf{Y}, \kappa)(\mathbf{a}) = \mathbf{y}$$

Besides retaining syntactic invariance and consistency from semantic conditioning, semantic conditioning at inference has other impactful properties that make it a suitable choice compared to semantic conditioning and regularization.

First, whereas semantic conditioning and semantic regularization rely on solving PQE to compute their loss module, semantic conditioning at inference only relies on solving MPE for its inference module. This has huge consequences on **computational tractability**: we demonstrate in Section 4.4 that there families of propositional formulas for which semantic conditioning at inference remains tractable while semantic conditioning and semantic regularization become intractable.

Second, integrating prior knowledge only at inference time offers more flexibility than integration during training. For instance, it can be used if prior knowledge is unavailable at training time (see for instance [18], which provides prior knowledge to an existing task of object detection) or susceptible to evolve. This is a particularly useful property in the era of **off-the-shelves** and **foundation** models [7], which are pre-trained on massive amounts of general data to then be applied in a multitude of heterogeneous downstream tasks, since task specific prior knowledge can not be integrated during most of the training process.

#### 4.4 Complexity and implementations

In this section we analyse the complexity of probabilistic reasoning, which is at the core of all techniques presented in the paper, and show that semantic conditioning at inference remains tractable on families of propositional formulas for which semantic conditioning and semantic regularization are not.

As mentioned in Section 2.3, all three neurosymbolic techniques defined in Section 4.2 internally rely on solving MPE and PQE problems. Unfortunately, MPE and PQE are NP-hard and #P-hard respectively in general (by a poly-time reduction from SAT and #SAT respectively). This implies that scaling these probabilistic neurosymbolic techniques to large classification tasks (i.e., tasks with a large number of variables) on arbitrary prior knowledge requires an exponential amount of computing resources (unless P = NP) and is therefore not realistic.

However, there are families of propositional formulas for which MPE and PQE problems can be solved tractably. This is the case of enumerable formulas (for which satisfying states can be listed in polynomial time) or bounded tree-width formulas for instance.

To go beyond tree-width, one approach that has become predominant in the literature is to use knowledge compilation to translate a propositional formula into a representation language that can solve probabilistic reasoning problems efficiently (i.e., in a time polynomial in the size of the compiled formula). Several fragments of boolean circuits [9] were identified as suitable compilation languages. Decomposable Negational Normal Form (DNNF) circuits can solve MPE problems in linear time and deterministic-DNNF (dDNNF) can additionally solve PQE problems in linear time. Sentential Decision Diagrams (SDD) [8] is a fragment of dDNNF that offers polynomial negation, conjunction and disjunction. Besides, [8] shows that a propositional formula  $\kappa$  in conjunctive normal form with  $k$  variables and a tree-width  $\tau$  has an equivalent compressed and trimmed SDD of size  $\mathcal{O}(k2^\tau)$ . Due to these properties, SDD has become a standard compilation language for probabilistic neurosymbolic systems [42, 1, 27]. Finally, [25] showed that acyclic simple path constraints can be compiled into an OBDD (a subset of SDD) of size polynomial in the number of edges in the graph, meaning that MPE and PQE problems can be solved tractably for propositional formulas that represent simple path constraints.

Finally, counting problems are known to be much harder in general than optimization problems [38]. For instance, MPE can be solved in polynomial time for formulas representing matching constraints (by reduction to finding a maximum weight-sum matching [15]), while PQE is still #P-hard [3]. As mentioned earlier in Section 4.3, this implies that semantic conditioning at inference is tractable on matching constraints while semantic conditioning and semantic regularization are not. In knowledge compilation, solving MPE only requires to compile the formula to a DNNF circuit while PQE requires in addition the target circuit to be deterministic, at the cost of larger compiled circuits and slower computations. Besides, this also reflects in the greater development and efficiency of optimization solvers compared to counting algorithms, which often makes semantic conditioning at inference easier to implement in practice (see below for simple path tasks). Any propositional formula in conjunctive normal form can also be converted to a binary linear program and semantic conditioning at inference can therefore exploit readily available libraries for mixed integer linear programming. Using this [31] presents an algorithm to approximate PQE using the Gumbel-max trick [32].

In all our experiments however, probabilistic reasoning computations brought no significant overhead on top of the neural network.

For categorical tasks, the number of valid states is enumerable in linear time, hence there are no complexity issues to implement all three techniques.

For hierarchical tasks, we implemented our own version <sup>1</sup> of [12], which uses a custom compilation algorithm to convert the proposi-

<sup>1</sup> Their code was not publicly available. Our code is attached in the supplementary materials and will be made publicly available in the final version.

tional formula into a minimal junction tree and then applies a sparse message passing procedure.

For simple path prediction task, we used the compilation technique in [25] alongside the SPL package [1] to implement semantic conditioning and semantic regularization. For semantic conditioning at inference, we simply adapted a shortest path solver from NetworkX [20] based on the Bellman-Ford algorithm [6].

## 5 Experiments

We showed in the previous section how semantic conditioning at inference was provably more accurate than independent multi-label classification. In this section, we illustrate to what extent this guarantee translates experimentally on a categorical task, two hierarchical tasks and a simple path prediction task. We also compare semantic conditioning at inference to existing techniques semantic conditioning and semantic regularization.

### 5.1 A new multi-scale methodology

Most papers in the field evaluate the benefits of their neurosymbolic technique on a single network size. Although informative, such a methodology paints a very limited picture of the benefits of the technique and leaves many questions unanswered. In particular, it does not allow to estimate how these benefits evolve when resources given to the system (*e.g.* network scale, dataset size, training time, etc.) increase. Besides, it is well known that performance is (marginally) increasingly more expensive to obtain in terms of resources [34], and appreciation of the value of prior knowledge and neurosymbolic techniques must take that into account. This leads to two main shortcomings. First, it provides limited insights into the sustainability of these benefits as the trend towards larger and deeper networks unfolds. Secondly, answering frugality-driven questions through this methodology is impossible.



To overcome those limitations, we developed a multi-scale evaluation methodology that studies the dependency between the performance of the neurosymbolic technique and the resources of the system. For each task, we selected a single architectural design that can be scaled to various sizes. We used a simple Convolutional Neural Network (CNN) [24] design on the MNIST dataset (Section 5.2.1) and the family of DenseNets [21] on all others (Sections 5.2.2 and 5.2.3). Then, across networks of different scales, we compared the performance of the three neurosymbolic techniques and an uninformed baseline (independent multi-label classification). We report the **exact accuracy** [1] (called coherent accuracy in [12]), *i.e.*, the share of instances which are well classified on all labels, as our evaluation metric. More details on the experimental setup (number of epochs, hyperparameters, etc.) are given in the supplementary materials.

Moreover, this methodology allows us to model the results points recorded for each technique [34] and therefore compute interesting metrics, such as asymptotic accuracy or frugality-driven like compression ratios (see the supplementary materials for more information).

## 5.2 Tasks

### 5.2.1 Categorical classification

We mentioned earlier (see Section 4) how categorical classification tasks could be framed as a multi-label classification with prior knowledge. MNIST is one of the oldest and most popular dataset in com-

puter vision and consists of small images of hand-written digits (*e.g.*  or ). Since its introduction in [24], it has been used as a *toy* dataset in many different settings. Likewise in neurosymbolic literature, many researchers used MNIST as a basis to build structured dataset compositionally (*e.g.* the PAIRS dataset in [28], the MNIST-Add dataset in [27, 5, 26, 39] or the Sudoku dataset in [4]).

### 5.2.2 Hierarchical classification

The Cifar-100 dataset [23] is composed of 60,000 images classified into 20 mutually exclusive super-classes (*e.g.* *reptile*), each divided into 5 mutually exclusive fine-grained classes (*e.g.* *crocodile*, *dinosaur*, *lizard*, *turtle*, and *snake*). While most papers only consider the categorical classification task arising from the 100 fine-grained classes, we keep all 120 classes to produce a multi-label classification task where prior knowledge captures mutual exclusion and the hierarchy between super and fine-grained classes.

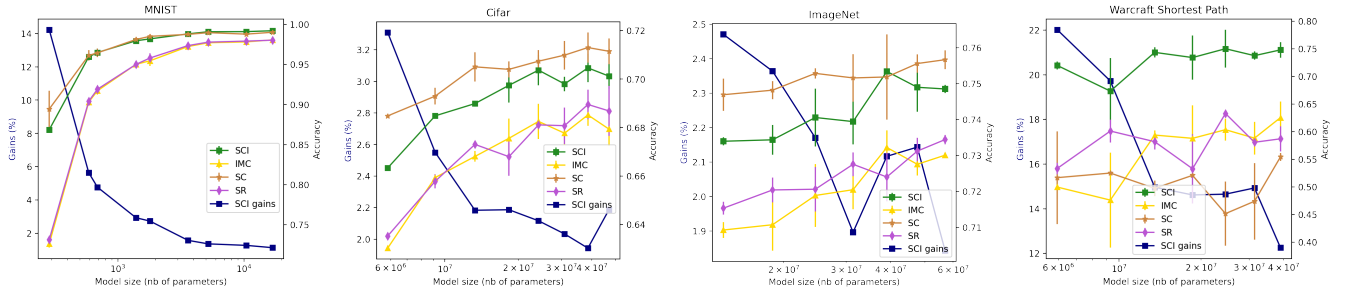
The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [35] is an image classification challenge which has become a standard benchmark in computer vision to compare performances of deep learning models. As of August 2014, ImageNet contained 14,197,122 annotated images organized into 21,841 synsets of the WordNet hierarchy [29], however standard image classification tasks often use a subset of those, usually 1,000 or 100 synsets. The WordNet hierarchy defines subsumption (or inclusion) between classes, and can be used in many ways to create a task of binary multi-label classification with prior knowledge.

For our experiments, we sample 100 classes from 1k ImageNet and add all their parent classes. We then prune classes that have only one parent class and one child class to avoid classes having identical sample sets. We thus obtain a dataset of ImageNet samples labeled on a hierarchy of 271 classes. Prior knowledge for this task includes the hierarchical knowledge coming from WordNet, as well as exclusion knowledge that we obtain by assuming two classes having no common descendants are mutually exclusive.

### 5.2.3 Simple path prediction

The Warcraft shortest path task [33, 43, 31, 1] uses randomly generated images of terrain maps from the Warcraft II tileset. Maps are build on a  $12 \times 12$  directed grid (each vertex is connected to all its *neighbors*) and to each vertex of the grid corresponds a tile of the tileset. Each tile is a RGB image that depicts a specific terrain, which has a fixed traveling cost. For each map, the label encodes the shortest s-t path (*i.e.*, a path from the upper-left to the lower-right corners), where the weight of the path is the sum of the traveling costs of all terrains (*i.e.*, grid vertices) on the path. The terrain costs are used to produce the dataset but are not provided during training nor inference. In the original dataset [33], output variables correspond to vertices in the grid and a state satisfies the simple path constraint if the vertices set to 1 constitute a simple s-t path.

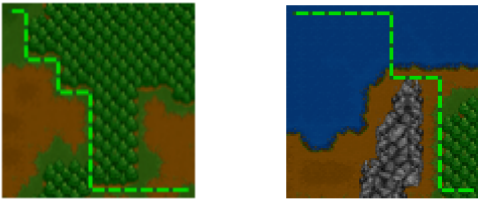
This representation comes with several issues. First, as noted in [1], the set of vertices ambiguously encode more than one path (because of cycles in the grid, there are several possible simple paths that go through the same vertices). Besides, computing MPE and PQE for simple path constraints on general directed graphs are respectively NP-hard and #P-hard [25]. To make this task tractable, [1] transforms the output space in the following way: edges of the grid are chosen as output variables instead of vertices and only simple paths with a maximal length of 29 (the maximal length found in the



**Figure 2.** From left to right: results at the last epoch on MNIST, Cifar, Imagenet and Warcraft shortest path. Each graph plots exact accuracy (right axis) for all four techniques against the size of the network. Additionally, we plot accuracy gains (*i.e.*, the accuracy gap with independent multi-label classification) for semantic conditioning at inference (left axis).

training set) are kept. This implies that the test set might not be consistent with the constraints since it might contain a path longer than 29 edges. Besides, such method would not scale to larger grids since the length of paths would grow exponentially.

In our experiments, we adopt a different approach. We keep the set of edges as our output variables, but we turn the grid into an acyclic graph by only connecting vertices to their right and lower neighbors. It is shown in [25] that acyclicity is a sufficient condition to make MPE and PQE tractable when variables are mapped to edges of the graph. This transformation allows us to scale to larger grids without an explosion of the computational cost. We recompute the labels for the new output space using the terrain costs.



**Figure 3.** Examples of Warcraft maps and their shortest paths [1]

### 5.3 Results and analysis

The results of our experiments are displayed on Figure 2, a graphical representation has been chosen over a tabular one to highlight how accuracy curves evolve as the network scales.

**Observation 1.** For all tasks, and across model scales, **semantic regularization brings little benefits** in terms of accuracy compared to independent multi-label classification.

This seems to indicate that semantic regularization benefits mainly come in a semi-supervised setting.

**Observation 2.** For MNIST, Cifar and ImageNet tasks, and across model scales, techniques based on the conditioned inference module  $\mathcal{J}_{sc}$  (*e.g.* **semantic conditioning** and **semantic conditioning at inference**) **significantly outperform** independent multi-label classification.

Besides, our results show that semantic conditioning at inference retains most of the performance gains (about 75%) of semantic conditioning, despite only integrating knowledge during inference.

**Observation 3.** For the Warcraft shortest path prediction task, **semantic conditioning at inference largely outperforms** all techniques across model scales. Moreover, beyond the two smaller networks, **semantic conditioning stagnates and deteriorates the performance** compared to independent multi-label classification.

**Observation 4.** Accuracy gains of semantic conditioning at inference **tend to decrease with the scale** of the neural network and converge towards a significantly positive value.

As the network scales and the performance of the neural classification system increases, performance improvements tend to slow down. In other terms, a 1 % accuracy improvement is much harder to obtain for larger networks than for smaller networks. Therefore, it is expected that accuracy gains brought by semantic conditioning and semantic conditioning at inference would decrease with the scale of the network. Interestingly however, our results show that these gains converge towards a significantly positive value, meaning that these techniques can improve performances on networks of all sizes.

## 6 Conclusion

In this paper, we introduce a formalism for supervised classification informed by propositional prior knowledge, define a new neurosymbolic technique called **semantic conditioning at inference** which integrates this prior knowledge during inference, and design a new methodology to evaluate the benefits of neurosymbolic techniques across model scales. To the best of our knowledge, this is the first neurosymbolic technique based on probabilistic reasoning which only impacts inference. We evaluate our technique alongside two existing techniques and show experimentally that **semantic conditioning at inference can improve the performances of a neural classification system on large datasets and on networks of all sizes**. Besides, we demonstrate that semantic conditioning at inference **preserves key properties** of semantic conditioning (*i.e.*, syntactic invariance and consistency) while **only working at inference**, making it **more flexible and tractable**. We show experimentally that our technique retains most of the performance gains of semantic conditioning on two hierarchical tasks and **largely outperforms** the other two techniques across model scales on a simple path prediction task.

Future directions for our work may include, amongst others, reproducing our experiments on more datasets, investigating the semi-supervised and partial-labels settings, or exploring alternative logics to represent prior knowledge. Finally, we assume throughout the paper that the knowledge is known *a priori*, which is often not the case in practice. Discovering the *structure* of the task at hand and training the model simultaneously is an important field of research.

## References

- [1] K. Ahmed, S. Teso, K.-W. Chang, G. den Broeck, and A. Vergari. Semantic Probabilistic Layers for Neuro-Symbolic Learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 29944–29959. Curran Associates, Inc., 2022.
- [2] K. Ahmed, E. Wang, K.-W. Chang, and G. Van den Broeck. Neuro-symbolic entropy regularization. In J. Cussens and K. Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 43–53. PMLR, 01–05 Aug 2022. URL <https://proceedings.mlr.press/v180/ahmed22a.html>.
- [3] A. Amarilli and M. Monet. Weighted counting of matchings in unbounded-treewidth graph families. URL <http://arxiv.org/abs/2205.00851>.
- [4] E. Augustine, C. Pryor, C. Dickens, J. Pujara, W. Y. Wang, and L. Getoor. Visual sudoku puzzle classification: A suite of collective neuro-symbolic tasks. In *International Workshop on Neural-Symbolic Learning and Reasoning*, 2022.
- [5] S. Badreddine, A. d. Garcez, L. Serafini, and M. Spranger. Logic Tensor Networks. *Artificial Intelligence*, 303:103649, 2022. ISSN 0004-3702. doi: <https://doi.org/10.1016/j.artint.2021.103649>. URL <https://www.sciencedirect.com/science/article/pii/S0004370221002009>.
- [6] R. Bellman. On a routing problem. 16(1):87–90. ISSN 0033-569X. URL <https://www.jstor.org/stable/43634538>. Publisher: Brown University.
- [7] R. Bommasani et al. On the opportunities and risks of foundation models. *ArXiv*, 2021. URL <https://crfm.stanford.edu/assets/report.pdf>.
- [8] A. Darwiche. SDD: A New Canonical Representation of Propositional Knowledge Bases. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.
- [9] A. Darwiche and P. Marquis. A knowledge compilation map. 17(1): 229–264. ISSN 1076-9757.
- [10] L. De Raedt, A. Kimmig, and H. Toivonen. ProbLog: a probabilistic prolog and its application in link discovery. In *Proceedings of the 20th international joint conference on Artificial intelligence, IJCAI'07*, pages 2468–2473, San Francisco, CA, USA, Jan. 2007. Morgan Kaufmann Publishers Inc.
- [11] J. Deng, O. Russakovsky, J. Krause, M. S. Bernstein, A. Berg, and L. Fei-Fei. Scalable multi-label annotation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '14*, pages 3099–3102. Association for Computing Machinery. ISBN 978-1-4503-2473-1. doi: [10.1145/2556288.2557011](https://doi.org/10.1145/2556288.2557011). URL <https://doi.org/10.1145/2556288.2557011>.
- [12] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-Scale Object Classification Using Label Relation Graphs. In *Computer Vision – ECCV 2014*, pages 48–64. Springer International Publishing, 2014. ISBN 978-3-319-10590-1.
- [13] M. Diligenti, M. Gori, and C. Sacca. Semantic-based regularization for learning and inference. *Artificial Intelligence*, 244:143–165, 3 2017. ISSN 00043702. doi: [10.1016/j.artint.2015.08.011](https://doi.org/10.1016/j.artint.2015.08.011).
- [14] T. Durand, N. Mehrasa, and G. Mori. Learning a deep ConvNet for multi-label classification with partial labels. URL <http://arxiv.org/abs/1902.09720>.
- [15] J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. 69B(1):undefined–undefined. ISSN 0022-4340. doi: [10.6028/jres.069b.013](https://doi.org/10.6028/jres.069b.013). URL <https://www.mendeley.com/catalogue/c0ac3d8d-0ca2-3d92-bd7e-d2570454b3e5/>. Number: 1 and 2.
- [16] F. Giannini, M. Diligenti, M. Maggini, M. Gori, and G. Marra. T-norms driven loss functions for machine learning. *Applied Intelligence*, 53(15):18775–18789, Feb. 2023. ISSN 0924-669X. doi: [10.1007/s10489-022-04383-6](https://doi.org/10.1007/s10489-022-04383-6). URL <https://doi.org/10.1007/s10489-022-04383-6>.
- [17] E. Giunchiglia and T. Lukasiewicz. Coherent hierarchical multi-label classification networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9662–9673. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/6dd4e10e3296fa63738371ec0d5df818-Abstract.html>.
- [18] E. Giunchiglia, M. C. Stoian, S. Khan, F. Cuzzolin, and T. Lukasiewicz. ROAD-R: the autonomous driving dataset with logical requirements. *Machine Learning*, 112(9):3261–3291, Sept. 2023. ISSN 1573-0565. doi: [10.1007/s10994-023-06322-z](https://doi.org/10.1007/s10994-023-06322-z). URL <https://doi.org/10.1007/s10994-023-06322-z>.
- [19] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *Proceedings of the 17th International Conference on Neural Information Processing Systems, NIPS'04*, pages 529–536, Cambridge, MA, USA, Dec. 2004. MIT Press.
- [20] A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught, and J. Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.
- [21] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, Honolulu, HI, July 2017. IEEE. ISBN 978-1-5386-0457-1. doi: [10.1109/CVPR.2017.243](https://doi.org/10.1109/CVPR.2017.243). URL <https://ieeexplore.ieee.org/document/8099726/>.
- [22] H. A. Kautz. The third ai summer: Aaai robert s. engelmore memorial lecture. *AI Mag.*, 43:93–104, 2022.
- [23] A. Krizhevsky. Learning multiple layers of features from tiny images, 2009.
- [24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2323, 1998. ISSN 00189219. doi: [10.1109/5.726791](https://doi.org/10.1109/5.726791).
- [25] A. Ledaguenel, C. Hudelot, and M. Khoudja. Complexity of probabilistic reasoning for neurosymbolic classification techniques, 2024.
- [26] J. Maene and L. De Raedt. Soft-unification in deep probabilistic logic. 36. URL [https://papers.nips.cc/paper\\_files/paper/2023/hash/bf215fa7fe70a38c5e967e59c44a99d0-Abstract-Conference.html](https://papers.nips.cc/paper_files/paper/2023/hash/bf215fa7fe70a38c5e967e59c44a99d0-Abstract-Conference.html).
- [27] R. Manhaeve, S. Dumančić, A. Kimmig, T. Demeester, and L. De Raedt. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence*, 298:103504, Sept. 2021. ISSN 0004-3702. doi: [10.1016/j.artint.2021.103504](https://doi.org/10.1016/j.artint.2021.103504). URL <https://www.sciencedirect.com/science/article/pii/S0004370221000552>.
- [28] G. Marra, F. Giannini, M. Diligenti, and M. Gori. Integrating learning and reasoning with deep logic models, 2020.
- [29] G. A. Miller. Wordnet. *Communications of the ACM*, 38:39–41, 11 1995. ISSN 15577317. doi: [10.1145/219717.219748](https://doi.org/10.1145/219717.219748). URL <https://dl.acm.org/doi/10.1145/219717.219748>.
- [30] B. R. Muller and W. Smith. A hierarchical loss for semantic segmentation. In *VISIGRAPP*, 2020. URL <https://api.semanticscholar.org/CorpusID:215791996>.
- [31] M. Niepert, P. Minervini, and L. Franceschi. Implicit MLE: Backpropagating through discrete exponential family distributions. In *Advances in Neural Information Processing Systems*, volume 34, pages 14567–14579. Curran Associates, Inc., 2021. URL [https://proceedings.neurips.cc/paper\\_files/paper/2021/hash/7a430339c10c642c4b2251756fd1b484-Abstract.html](https://proceedings.neurips.cc/paper_files/paper/2021/hash/7a430339c10c642c4b2251756fd1b484-Abstract.html).
- [32] G. Papandreou and A. L. Yuille. Perturb-and-MAP random fields: Using discrete optimization to learn and sample from energy models. In *Proceedings of the 2011 International Conference on Computer Vision, ICCV '11*, pages 193–200. IEEE Computer Society, 2011. ISBN 978-1-4577-1101-5. doi: [10.1109/ICCV.2011.6126242](https://doi.org/10.1109/ICCV.2011.6126242). URL <https://doi.org/10.1109/ICCV.2011.6126242>.
- [33] M. V. Pogančić, A. Paulus, V. Musil, G. Martini, and M. Rolinek. Differentiation of Blackbox Combinatorial Solvers. Sept. 2019. URL <https://openreview.net/forum?id=BkevoJSYPB>.
- [34] J. S. Rosenfeld, A. Rosenfeld, Y. Belinkov, and N. Shavit. A constructive prediction of the generalization error across scales. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL <https://openreview.net/forum?id=ryenvpEKDr>.
- [35] O. Russakovsky et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 12 2015. ISSN 15731405. doi: [10.1007/s11263-015-0816-y](https://doi.org/10.1007/s11263-015-0816-y).
- [36] S. Russell and P. Norvig. *Artificial Intelligence A Modern Approach (4th Edition)*. Pearson Higher Ed, 2021.
- [37] M. Seeger. Learning with labeled and unlabeled data. 2000.
- [38] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991. ISSN 0097-5397. doi: [10.1137/0220053](https://doi.org/10.1137/0220053). URL <https://epubs.siam.org/doi/10.1137/0220053>. Publisher: Society for Industrial and Applied Mathematics.
- [39] E. van Krieken, T. Thanapalasingam, J. Tomczak, F. van Harmelen, and A. Ten Teije. A-NeSI: A scalable approximate method for probabilistic neurosymbolic inference. 36. URL [https://proceedings.neurips.cc/paper\\_files/paper/2023/hash/4d9944ab3330fe6af8efb9260aa9f307-Abstract-Conference.html](https://proceedings.neurips.cc/paper_files/paper/2023/hash/4d9944ab3330fe6af8efb9260aa9f307-Abstract-Conference.html).
- [40] L. von Rueden et al. Informed Machine Learning – A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):614–633, Jan. 2023. ISSN 1558-2191. doi: [10.1109/TKDE.2021.3079836](https://doi.org/10.1109/TKDE.2021.3079836). Conference Name: IEEE Transactions on Knowledge and Data Engineering.
- [41] W. Wang, Y. Yang, and F. Wu. Towards data-and knowledge-driven artificial intelligence: A survey on neuro-symbolic computing, 2023. URL <https://arxiv.org/abs/2210.15889>.
- [42] J. Xu, Z. Zhang, T. Friedman, Y. Liang, and G. V. D. Broeck. A semantic loss function for deep learning with symbolic knowledge. In



*35th International Conference on Machine Learning, ICML 2018*, volume 12, pages 8752–8760. International Machine Learning Society (IMLS), 2018. ISBN 9781510867963.

- [43] Z. Yang, A. Ishay, and J. Lee. NeurASP: Embracing neural networks into answer set programming. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 1755–1762. International Joint Conferences on Artificial Intelligence Organization. ISBN 978-0-9992411-6-5. doi: 10.24963/ijcai.2020/243.