Highlights

Defining Neurosymbolic AI

- Research highlight 1
- Research highlight 2

Defining Neurosymbolic AI

Abstract

Neurosymbolic AI focuses on integrating learning and reasoning, in particular, on unifying logical and neural representations. Despite the existence of an alphabet soup of neurosymbolic AI systems, the field is lacking a generally accepted definition of what neurosymbolic models and inference really are. We introduce a formal definition for neurosymbolic AI that makes abstraction of its key ingredients. More specifically, we define neurosymbolic inference as the computation of an integral over a product of a logical and a belief function. We show that our neurosymbolic AI definition makes abstraction of key representative neurosymbolic AI systems.

Keywords: Neurosymbolic AI

1. Introduction

Neurosymbolic AI (NeSy) is a term for AI models that combine more traditional symbolic AI techniques with the latest advances in deep learning. It integrates raw neural and numerical data processing with symbolic reasoning and background knowledge. The advantages of this combination have already been demonstrated in applications such as providing safety guarantees [35], learning from distant signals and supervision by deduction [2], and more [12, 19, 29]. Neurosymbolic AI is attracting a lot of attention [18, 6, 13, 11, 32, 9, 3, 30, 8] and has been termed *"the most promising approach to a broad AI"* by Hochreiter [18] and the *"3rd wave in AI"* by Garcez and Lamb [13]. It is mentioned as an innovation trigger on Gartner's hype cycle[1] and there are now dedicated journals[2], conferences[3], and summer schools[4] devoted to neurosymbolic AI.

Despite the wide interest, the term neurosymbolic AI is used for describing many different types of integrations of neural and symbolic AI systems. For instance, Henry Kautz describes six different types of such integrations [20], some requiring tighter interfaces between the neural and the symbolic component, others looser ones. As Garcez and Lamb [13], we will focus on the original and stricter interpretation of the term neurosymbolic AI, which Garcez and Lamb describe as *"research that integrates in a principled way neural network-based*

[1] https://www.gartner.com/en/articles/hype-cycle-for-artificial-intelligence
[2] https://neurosymbolic-ai-journal.com/
[3] https://2025.nesyconf.org/
[4] https://neurosymbolic.github.io/nsss2024/

learning with symbolic knowledge representation and logical reasoning". This is also the dominant view in the Neurosymbolic AI Journal and Conference. Within this view there exist numerous models, systems and techniques that integrate logic with neural network-based approaches, see for instance [24, 14, 6, 17] for overviews. However, the focus in the field is very much on designing bespoke systems that score best on the latest benchmarks, which results in an alphabet-soup of systems. This comes at the expense of understanding the underlying principles and commonalities that these systems share, which is significantly hindering progress in the field. What is lacking is a commonly agreed formal definition and framework for specifying, comparing and developing neurosymbolic AI models and problems. It is precisely this gap that this paper wants to bridge.

More specifically, we contribute formal definitions of a neurosymbolic model and neurosymbolic inference. These definitions are based on the observation that the vast majority of neurosymbolic AI models combine *logic* with *beliefs*. The term logic refers to the wide variety of logics that are used in neurosymbolic AI ranging from Boolean logic to first-order and fuzzy logic, as well as their combinations. The term belief refers to a weighting component that many neurosymbolic AI models [22, 34, 26, 36] derived from statistical relational AI models [10] use. Within our framework, neurosymbolic inference can be cast as aggregation (or marginalisation) of a product of a logic and a belief function. Our definitions provide a semantic framework for NeSy models that clarifies their components and the way they interact. We will show that many representative classes of NeSy models and tasks, including those based on probabilistic logic [22, 36], fuzzy logic [4], or soft logic [26], can be cast within our definitions by instantiating the logic, the belief function and the neural networks appropriately. As a consequence, our definitions can be used to relate, compare and develop different NeSy models in a principled manner, as well as to study fundamental properties of NeSy.

2. Logic as the symbol level

In Garcez and Lamb's perspective on neurosymbolic AI, the symbol level is viewed as symbolic knowledge representation and logical reasoning. We will therefore focus on using logical languages, although the proposed definitions in principle apply to other formal languages and automata. We will also allow for a wide range of semantics for these languages, in order to support Boolean, fuzzy and other logics.

More formally, a language is a set of sentences over a set $\mathcal{V} = \{V_i\}_{i \in I}$ of variables with domains D_i that can be embedded in $\mathbb{R}$ and interact with operators O.[5] An interpretation $\omega : \mathcal{V} \to \mathbb{R}$ is an assignment of values to the variables and we use $\Omega = \mathbb{R}^{\mathcal{V}}$ tot denote the set of all possible interpretations over the variables $\mathcal{V}$. Given an interpretation $\omega \in \Omega$ and a sentence $\varphi \in L$, the *semantics*

[5]The assumption of having $\mathbb{R}$ as a shared domain is made to simplify notation and can easily be enforced, e.g. the domain of a discrete variable is often represented as integers.

$\mu : L \times \Omega \to S$ of the language L maps φ to a semantic value $\mu(\varphi, \omega)$ in the interpretation ω, which we will often denote as $\varphi(\omega)$ for brevity. The set S of semantic values is also assumed to be embeddable in $\mathbb{R}^+$.[6]

Note that variables can be separated into two distinct classes based on their domain D_i. If D_i is equal to the set of semantic values S, then the variable V_i can be semantically interpreted and we will call it an *atom variable*. If it is not, then V_i does not have a direct semantic interpretation and it will instead be used as an argument to an operator. Hence, we will call the latter *argument variables*.

We illustrate the different choices of language and semantics using both fuzzy [37, 27] and Boolean semantics for propositional logic sentences.

Example 2.1 (Boolean propositional logic)**.** Propositional logic is the language that consists of sentences over a set $\mathcal{V}$ of only atom variables called *propositions* that can be connected with logical connectives such as $\implies$ and $\vee$. For example, the sentence φ

$$\mathsf{happy} \implies (\mathsf{coffee} \vee \mathsf{publication}), \tag{2.1}$$

states that happiness h is only possible when having either a coffee c or a publication p. Propositional logic can be equipped with a Boolean semantics by using $S = \{0, 1\}$ as domain for all propositions denoting false and true and considering the interpretations ω as mappings from $\mathcal{V}$ to S. The Boolean semantics μ_B of propositional logic formulae then follow inductively. For instance, the above formula φ evaluates to $\varphi_B(\omega) = 1$ for the interpretation

$$\omega_B(a) = \begin{cases} 1 & \text{if } a \in \{\mathsf{h}, \mathsf{c}\}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

Example 2.2 (Fuzzy propositional logic)**.** Propositional logic can also be equipped with a fuzzy semantics by setting the set S of semantic values to the real unit interval $[0, 1]$ and using fuzzy operators such as the continuous T-norms [16] to inductively define its semantics μ_F.

For instance, if one uses the Łukasiewicz T-norm $\max(0, x + y - 1)$ for the conjunction and its corresponding T-conorm $\min(1, x + y)$ for the disjunction, the formula φ evaluates to $\varphi_F(\omega) = \min(1, 1 - \omega(\mathsf{h}) + \min(1, \omega(\mathsf{c}) + \omega(\mathsf{p}))) = 1$ for the interpretation

$$\omega_F(a) = \begin{cases} 0.5 & \text{if } a \in \{\mathsf{c}, \mathsf{p}\}, \\ 1 & \text{otherwise.} \end{cases} \tag{2.3}$$

Example 2.3 (Linear SMT logic)**.** The language of linear SMT logic is comprised of sentences over a set of only argument variables that use linear arithmetic operators to form linear arithmetic comparisons that can be connected with

[6]This assumption also simplifies notation and is not a hard one, e.g. it is common to represent Boolean truth values $\top$ and falsehood $\perp$ as 1 and 0.

any logical operator. For example, one can rewrite the propositional sentence of Equation 2.1 as the linear SMT formula

$$\texttt{h = 1} \implies \texttt{(c + p > 0)}, \tag{2.4}$$

where h, c and p are argument variable with domain $\{0, 1\}$. SMT logic can also be given a Boolean semantics by mapping arithmetic comparisons to the set $S = \{0, 1\}$ by choosing 1 for interpretations, i.e. assignments of argument variables, that satisfy the comparison according to arithmetic and 0 otherwise.

Numerous other logics exist, such as first-order and temporal logics, and their semantics are also defined in terms of assigning values to variables or sequences of variables (for temporal logics). Similar analyses can be made for automata and other formal languages.

Logical inference can be viewed as inferring whether there are interpretations that satisfy certain constraints w.r.t. their semantic values. For instance, SAT-solvers address the question whether there exists an interpretation ω that satisfies a formula φ, i.e. that satisfies $\varphi_B(\omega) = 1$. In fuzzy logic, one might be interested in considering only those interpretations ω that satisfy $\varphi_F(\omega) > \tau$. For these reasons, it will be convenient to introduce logic functions.

Definition 2.4. (Logic function) A *logic function* l is a function that takes a formula φ and interpretation ω and returns a non-zero semantic value only when the value of $\varphi(\omega)$ is contained in a desired subset S_l of the semantic values S. That is, it is any function $l : L \times \Omega \to S$ with $l(\varphi, \omega) = 0$ if $\varphi(\omega) \notin S_l$. We call the set S_l the *selection values* of the logic function.

The intuition behind a logic function is that it outputs desired semantic values for selected interpretations of interest based on a logical formula. For most NeSy AI systems, especially those based on Boolean logic, $l(\varphi, \omega) = \varphi(\omega)$. It is only when working with both thresholds and fuzzy logic that more complex logic functions might be necessary, such as $l(\varphi_F, \omega) = [\![\varphi_F(\omega) > \tau]\!]$, where $[\![]\!]$ denotes the Iverson brackets that evaluate to 1 if its argument evaluates to true, and yields the value 0 otherwise.

3. Towards Neurosymbolic Models and Inference

Neurosymbolic AI models extend logic with neural networks and are, as we will show, typically based on two components: a (possibly fuzzy) logic and a (possibly probabilistic) neural belief [24]. While the semantic value $\varphi(\omega)$ of a logical formula in an interpretation is captured by the semantic function $\mu(\varphi, \omega)$, the belief component can be captured by the belief function $b_{\boldsymbol{\theta}}(\varphi, \omega)$. The belief $b_\theta(\varphi, \omega)$ can be interpreted as the weight indicating the degree of belief that the interpretation ω satisfies the formula φ. It generally takes the form of a parametrised function $b_{\boldsymbol{\theta}} : L \times \Omega \to \mathbb{R}$ with parameters $\boldsymbol{\theta}$ that takes an assignment of variables and a sentence and outputs the belief. Luc: insert statement as to how this fits with propositional variables

Example 3.1 (Parametrising a Boolean propositional sentence)**.** Consider again the sentence $\mathsf{h} \implies (\mathsf{c} \vee \mathsf{p})$ from Example 2.1. Now assume there is a neural network that takes as input an image taken from a camera in your local mathematics department. As output, the network returns three probabilities, one probability $p_{\boldsymbol{\theta},i}$ for each variable $i \in \{\mathsf{h}, \mathsf{c}, \mathsf{p}\}$. Together, assuming h, c and p are independent, these probabilities can be combined into a simple probabilistic belief function $b_{\boldsymbol{\theta}}(\varphi, \omega)$ by taking the product of the probabilities, i.e.

$$b_{\boldsymbol{\theta}}(\varphi, \omega) = \prod_{i \in \{\mathsf{h},\mathsf{c},\mathsf{p}\}} p_{\boldsymbol{\theta},i}^{\omega(i)} \cdot (1 - p_{\boldsymbol{\theta},i})^{1-\omega(i)}. \tag{3.1}$$

Definition 3.2 (Neurosymbolic model)**.** A neurosymbolic AI model $(L, \mu, \Omega, b_{\boldsymbol{\theta}})$ consists of a logical language L with a semantics μ over interpretations Ω and a belief function $b_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$.

Neurosymbolic AI models are used to perform inference. We view inference in a neurosymbolic model $(L, \mu, \Omega, b_{\boldsymbol{\theta}})$ as computing the integral graphically illustrated in Figure 1. Neurosymbolic inference can be formally defined through neurosymbolic functionals.

Definition 3.3 (Neurosymbolic Inference)**.** Given a neurosymbolic model $(L, \mu, \Omega, b_\theta)$ and a logic function l, neurosymbolic inference is defined as computing the result of the following *neurosymbolic functional*

$$F_{\boldsymbol{\theta}}(\varphi) = \int_{\Omega} l(\varphi, \omega)\; b_{\boldsymbol{\theta}}(\varphi, \omega)\; \mathrm{d}\omega. \tag{3.2}$$

Importantly, this definition leads to precise conditions under which neurosymbolic inference is well-defined. The conditions trivially follow from the definition of the Lebesgue integral (Appendix A).

Proposition 3.4 (Well-defined neurosymbolic inference)**.** Let $(L, \mu, \Omega, b_{\boldsymbol{\theta}})$ be a neurosymbolic AI model and l a logic function. If $(\Omega, \Sigma_\Omega, \mathrm{d}\omega)$ is a measure space for which the logic function l and belief function $b_{\boldsymbol{\theta}}$ are measurable, then neurosymbolic inference is well-defined.

The choice of how to parametrise the belief $b_{\boldsymbol{\theta}}$ is completely free and does not necessarily have to involve neural networks. In fact, if one foregoes the use of neural networks and considers probabilistic beliefs, then our definition of a neurosymbolic model reduces to a definition for statistical relational AI (StarAI) [10] models. Consequently, our view on neurosymbolic AI inference immediately connects to inference in StarAI as well, showing the use of a unifying and formal definition.

The question of how to perform learning from the quantities inferred by a neurosymbolic functional can be answered in many different ways. In settings where the belief is parametrised by neural networks, one can define a loss function in terms of the neurosymbolic functional and learn via backpropagation of this loss. In other settings, such as in StarAI, different approaches to learning such

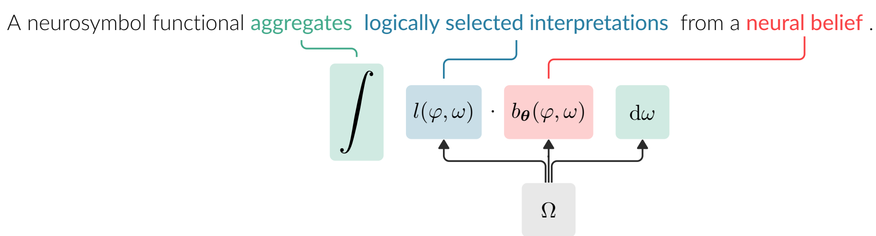

Figure 1: The main intuition behind neurosymbolic inference. Note how the interpretations Ω of the logical language form the interface between neural and symbolic components.

Table 1: How to recover popular neurosymbolic frameworks as neurosymbolic inference. (B) indicates the semantics are Boolean in the sense that they are based on true and false values while (F) similarly indicates fuzzy semantics. The asterisk * expresses that probabilistic parametrisations are allowed to be unnormalised.

System	Language	Semantics	Belief function	Logic function
DeepProbLog	Logic programs	Well-founded (B)	Probabilistic	Boolean satisfaction
NMLN	First-order logic	Boolean (B)	Probabilistic*	Boolean satisfaction
SPL	Propositional logic	Boolean (B)	Probabilistic circuit	Boolean satisfaction
NeurASP	Logic programs	Stable models (B)	Probabilistic	Boolean satisfaction
NeuPSL	Logic programs	Łukasiewicz (F)	Probabilistic*	Expected fuzzy values
LTN	First-order logic	Fuzzy (F)	Embedded formula	Fuzzy satisfaction
SBR	First-order logic	Fuzzy (F)	Point prediction	Fuzzy satisfaction

as expectation maximisation can also be used. In general, our framework does not impose any restrictions on how to perform learning. We only define the semantics of NeSy models and of inference, not of learning, as is usual in semantic frameworks.

4. Neurosymbolic inference unify neurosymbolic AI

We will now show that our definitions of neurosymbolic models and inference unifies many prominent neurosymbolic AI frameworks. These frameworks can be characterised based on their language, semantics and parametrisation (Table 1). We will mainly separate systems based on their Luc: boolean of fuzzy? semantics.

Add extended table to appendix with all systems we know of. Only systems in this table will be discussed in detail.

4.1. Neurosymbolic AI with Boolean semantics

Boolean logic is fundamental to computer science and is also the foundation of a series of neurosymbolic systems with a *probabilistic interpretation*, such as DeepProbLog [21], Neural Markov Logic Networks (NMLN) [23], Semantic Probabilistic Layers (SPL) [1] and NeurASP [36]. All of these systems differ in their choice of language, parametrisation or implementation of Boolean semantics, e.g. stable model semantics [15] or well-founded semantics [31]. However, they are similar in that they are all based on computing probabilities of sentences being true or false. Returning to our running example, each of the systems can compute the probability of the sentence "happiness is only possible when having coffee or a publication" encoded in their respective languages.

Example 4.1 (Probabilistic Boolean neurosymbolic AI)**.** Assuming an independent factorisation $b_{\boldsymbol{\theta}}$ (Equation 3.1), the probability of the sentence $\mathsf{h} \implies (\mathsf{c} \vee \mathsf{p})$ being true is by definition

$$\int_{\mathbb{B}^3} \mu(\mathsf{h} \implies (\mathsf{c} \vee \mathsf{p}), \omega) \prod_{i \in \{\mathsf{h},\mathsf{c},\mathsf{p}\}} p_{\boldsymbol{\theta},i}^{\omega(i)} \cdot (1 - p_{\boldsymbol{\theta},i})^{1-\omega(i)} \, \mathrm{d}\mathsf{h}\mathrm{d}\mathsf{c}\mathrm{d}\mathsf{p}, \tag{4.1}$$

where $\mathrm{d}\mathsf{h}$, $\mathrm{d}\mathsf{c}$ and $\mathrm{d}\mathsf{p}$ are all binary counting measures that enumerate the 8 possible binary interpretations. That is, we have neurosymbolic inference with the Boolean semantics μ as logic function and a probabilistic belief function.

Each of the different Boolean neurosymbolic systems would perform this exact computation, but with their own language and semantics. NMLN and SPL would directly encode the sentence in first-order logic with the same Boolean semantics as above. For DeepProbLog and NeurASP, the sentence $\mathsf{h} \implies (\mathsf{c} \vee \mathsf{p})$ would be encoded as a logic program with its corresponding well-founded or stable models semantics.

In general, neurosymbolic systems based on Boolean semantics perform inference according to Definition 3.3.

Claim 4.2. Inference in typical neurosymbolic systems based on Boolean semantics corresponds to neurosymbolic inference of the form

$$\int_{\Omega_B} l(\varphi, \omega) \; b_{\boldsymbol{\theta}}(\varphi, \omega) \mathrm{d}\omega, \tag{4.2}$$

where $\Omega_B = \mathbb{B}^{\mathcal{V}}$ is the set of all functions from atom variables $\mathcal{V}$ to Boolean semantic values $\mathbb{B} = \{0, 1\}$.

Argument. We show that this statement holds for the case of 1) DeepProbLog, SPL, NeurASP and 2) NMLNs. The foundational inference task in these systems is computing the probability that a sentence φ is true, i.e.

$$\mathbb{P}(\varphi) = \int_{\Omega_B} \varphi_B(\omega) b_{\boldsymbol{\theta}}(\varphi, \omega) \; \mathrm{d}\omega. \tag{4.3}$$

Hence, the logic function l for DeepProbLog, SPL, NeurASP and NMLNs is equal to the Boolean value $\varphi_B(\omega)$ of the sentence φ in the interpretation ω. The belief function $b_{\boldsymbol{\theta}}$ for all systems has to be a probability distribution, yet their form differs per system. DeepProbLog and NeurASP choose an independently factorising probability distribution as belief. That is, their belief function is

$$b_{\boldsymbol{\theta}}(\varphi, \omega) = \prod_{a \in A} p_{\boldsymbol{\theta},a}(\omega(a)). \tag{4.4}$$

SPL allows to parametrise the belief as a conditional probabilistic circuit [28] that needs to be compatible with the logical formula φ. NMLNs see the sentence φ as a first-order theory $\bigwedge_{i=1}^{N} \varphi_i$ consisting of N sentences. Their belief function is then constructed as the normalised exponentiated sum

$$b_{\boldsymbol{\theta}}(\varphi, \omega) = \frac{1}{Z} e^{\sum_{i=1}^{N} \lambda_{\boldsymbol{\theta},i} \cdot \varphi_{i,B}(\omega)} = \frac{1}{Z} \prod_{i=1}^{N} e^{\lambda_{\boldsymbol{\theta},i} \cdot \varphi_{i,B}(\omega)}, \tag{4.5}$$

where each $\lambda_{\boldsymbol{\theta},i}$ is a parametrised weight. □

In the probabilistic setting where the belief $b_{\boldsymbol{\theta}}(\varphi, \omega)$ is a probability distribution over the set of interpretations Ω, neurosymbolic inference becomes an instance of either weighted model counting (WMC) [7] or weighted model integration (WMI) [5] depending on whether Ω is finite or infinite.

4.2. Neurosymbolic AI with fuzzy semantics

Fuzzy semantics for logical languages [27] has enjoyed much interest as a continuous, more fine-grained alternative to the traditional Boolean semantics. While Boolean semantics is a two-valued semantics based on absolute truth and falsehood, fuzzy logic is an infinite-valued semantics [27] Luc: does not have to be continuousLennert: Here continuous in the sense of having $\mathbb{R}$ as domain. Should I say "uncountable degree of truth?" that expresses a continuous degree of truth by mapping symbols and sentences to the real unit interval. For neurosymbolic AI, the continuous nature of fuzzy semantic values can result in a differentiable notion of satisfiability that makes the integration with neural networks easier. However, it does lead to more diverse computations as different systems can be interested in different restrictions of the fuzzy values of a sentence.

Example 4.3 (Fuzzy neurosymbolic AI). Luc: tough example Many fuzzy neurosymbolic systems only compute the fuzzy value of a sentence given a single fuzzy interpretation. For example, while Logic Tensor Networks (LTN) proposes a construction to embed constants, variables, and functions as tensors or tensor operations, they end up with fuzzy predicates that use fuzzy logic operators to yield the fuzzy value of a sentence. Put differently, LTN proposes an intricate way of parametrising a belief $b_{\boldsymbol{\theta}}$ for a single fuzzy interpretation $\omega_{\boldsymbol{\theta}}$ by assigning a single fuzzy value to each of the atom variables. In case of our running example with the Łukasiewicz T-norm, the belief $b_{\boldsymbol{\theta}}$ of LTN would be a Dirac delta function that gives a single fuzzy value for h, c and p while ignoring all the other possible fuzzy interpretations in the space $\Omega_F = [0, 1]^3$, leading to neurosymbolic inference of the form

$$\int_{\Omega_F} \min(1, 1 - \omega(\mathsf{h}) + \min(1, \omega(\mathsf{c}) + \omega(\mathsf{p})))\delta(\omega - \omega_{\boldsymbol{\theta}}) \, \mathrm{d}\omega = \varphi_F(\omega_{\boldsymbol{\theta}}). \tag{4.6}$$

This expression uses fuzzy evaluation as logic function and collapses to the fuzzy value $\varphi_F(\omega_{\boldsymbol{\theta}})$ because of the Dirac delta function δ.

In general, our definition of neurosymbolic inference encompasses inference in typical fuzzy neurosymbolic systems. While covering LTN and SBR inference requires a slightly more complex rewrite of their inferred quantities, rewriting exposes connections to other fuzzy systems like NeuPSL.

Claim 4.4. Inference in typical neurosymbolic systems based on fuzzy semantics is neurosymbolic inference of the form

$$\int_{\Omega_F} l(\varphi, \omega) b_{\boldsymbol{\theta}}(\varphi, \omega) \, \mathrm{d}\omega, \tag{4.7}$$

where $\Omega_F = [0,1]^{\mathcal{V}}$ is the set of all functions from the atom variables $\mathcal{V}$ to fuzzy values in $[0,1]$.

Argument. We prove this claim for 1) logic tensor networks (LTN), semantic-based regularisation (SBR) and 2) neural probabilistic soft logic (NeuPSL).
LTN and SBR both compute the fuzzy value $\varphi_F(\omega_{\boldsymbol{\theta}})$ of a sentence φ in a single parametrised fuzzy interpretation $\omega_{\boldsymbol{\theta}}$. Only considering a single fuzzy interpretation corresponds to choosing a belief function that is a Dirac delta distribution, i.e. $b_{\boldsymbol{\theta}}(\varphi, \omega) = \delta(\omega - \omega_{\boldsymbol{\theta}})$. Indeed, we can use the collapsing property of the Dirac delta distribution δ to write

$$\varphi_F(\omega_{\boldsymbol{\theta}}) = \int_{\Omega_F} \varphi_F(\omega) \delta(\omega - \omega_{\boldsymbol{\theta}}) \, \mathrm{d}\omega. \tag{4.8}$$

Hence, LTN and SBR both have the fuzzy value $\varphi_F(\omega)$ as logic function and a Dirac delta as belief function.
NeuPSL sets the belief function to be the probability distribution

$$b_{\boldsymbol{\theta}}(\varphi, \omega) = \frac{1}{Z} e^{\sum_{i=1}^{N} \lambda_{\boldsymbol{\theta},i} \cdot \varphi_{i,F}(\omega)} = \frac{1}{Z} \prod_{i=1}^{N} e^{\lambda_{\boldsymbol{\theta},i} \cdot \varphi_{i,F}(\omega)}, \tag{4.9}$$

similarly to NMLNs, but with fuzzy semantics. Its choice of logic function changes from task to task, but NeuPSL generally computes fuzzy expected values. For instance, the usual expected fuzzy value would use fuzzy satisfaction $\varphi_F(\omega)$ as logic function. □

Note how LTN and SBR use a very simple belief function that only parametrises a single fuzzy interpretation while NeuPSL parametrises a belief over all fuzzy interpretations.

Example 4.5 (Probabilistic fuzzy neurosymbolic AI)**.** Systems like NeuPSL relax the hard, Boolean semantic values of atomic expressions to soft, fuzzy values in Łukasiewicz logic and define a probability distribution $p(\varphi, \omega)$ over the space of fuzzy interpretations. This construction allows computing *fuzzy expectations*, e.g. the expectation of the fuzzy value of the sentence $\mathsf{h} \implies (\mathsf{c} \vee \mathsf{p})$

$$\mathbb{E}_{\omega \sim p(\varphi,\omega)} \left[\varphi_F(\omega)\right] = \int_{\Omega_F} \min(1, 1 - \omega(\mathsf{h}) + \min(1, \omega(\mathsf{c}) + \omega(\mathsf{p}))) p(\varphi, \omega) \, \mathrm{d}\omega. \tag{4.10}$$

This quantity can then be used to optimise the fuzzy value of a sentence *in expectation* instead of only relying on a point estimate as LTN or SBR does.

Given the example of NeuPSL, it seems fuzzy neurosymbolic systems can become more expressive by parametrising the continuum of fuzzy interpretations. Indeed, instead of parametrising a probability distribution over all fuzzy interpretations, one can use any other expressive belief function that covers more than one fuzzy interpretations. For instance, to maintain a completely fuzzy approach, one could turn the set Ω_F of fuzzy interpretations itself into a fuzzy set by parametrising a membership function $m : \Omega \to [0, 1]$ that corresponds to defining a fuzzy belief function. Luc: start with the example of NeuPSL and then go for the implciations ... more digestable? Lennert: Done, this does seem better.

4.3. Limitations

Our definition of neurosymbolic inference (Definition 3.3) makes certain assumptions to simplify the exposition. For one, Equation 3.2 always integrates over the entire space of interpretations, which does not cover marginal maximum a posteriori (MMAP) tasks where only part of the interpretations is integrated out while another is maximised over. Moreover, other inference tasks might require nesting or composing different instances of Equation (3.2). For instance, conditional inference in probabilistic neurosymbolic systems, e.g. SPL and DeepProbLog, first compute the normalisation constant to then define conditional belief functions. Another example is the weighted maxSAT task where a weighted sum of satisfied clauses has to be maximised. This task can be seen as a maximisation operation on top of a series of separate instances of Equation 3.2. Covering all of these different cases would go beyond the scope of this paper, as we focus on a simple definition of the most foundational neurosymbolic inference tasks. Such a discussion is left for future work.

5. Related work

This is not the first attempt to arrive at a synthesis and a framework for neurosymbolic AI. For instance, Odense and Garcez [25] introduce a semantic framework for *encoding* logics into neural networks, and have a similar motivation as our work. However, their emphasis is on the necessary conditions under which a class of neural networks and logical systems can be said to be semantically equivalent. That is, any specific neural network can be encoded as a logical theory and the other way around. This is in line with Henry Kautz's category $\mathbf{Neural}_{\mathbf{Symbolic}}$ to produce a neural network from logical rules. Our semantics focuses more on making the logical and belief functions explicit, rather being implicit about the neural network architecture, which is thus more in line with Henry Kautz's category **Neural** | **Symbolic** in which both the logical and neural components remain and one is not reduced to the other.

Another related work is ULLER [33], which proposes a unified language for learning and reasoning. It aims at the "frictionless sharing of knowledge" across neurosymbolic systems and is intended as an interface language, or even interface system, for contemporary neurosymbolic AI systems. Unlike our approach, ULLER is built upon a fixed first order logic and assigns a concrete semantics for Boolean, fuzzy and probabilistic instances. In contrast, our framework is neither looking for a lingua franca nor a unifying system for neurosymbolic AI, but rather focuses on defining a wide variety of neurosymbolic models and inference tasks in a mathematically sound way.

Other noteworthy approaches include van Bekkum et al. [30], who show how to combine and visualize specific design patterns of learning and reasoning architectures, Dash et al. [8] who characterise NeSy systems by input formats and loss functions, and Marra et al. [24] who devise various dimensions of neurosymbolic and statistical relational AI systems on which our definition builds. While these are important developments that can eventually lead to an extensive taxonomy of neurosymbolic AI systems, our definitions identify the essential concepts of neurosymbolic models and show how these concepts define abstract neurosymbolic inference tasks.

insert https://arxiv.org/pdf/2303.10

6. Conclusion

Lennert: Here we already summarise the advantages of the definition, right? Or would we also put it somewhere earlier and clearer too?

Motivated by the wide range of existing neurosymbolic AI models and approaches, which appear quite different on the surface level, we proposed a general and in a sense unifying definition of neurosymbolic AI systems that integrate neural networks with logics. In our view, neurosymbolic inference consists of computing an integral over a product of a logical and a belief function. We provided evidence that our framework is general in that is makes abstraction of prominent contemporary systems such as LTNs, NeuPSL, SBR, SPL, DeepProbLog, NeurASP and NMLNs.

We believe that our definition will be useful for for developing both the theory of neurosymbolic AI by providing a computational framework for designing, evaluating and comparing different neurosymbolic AI systems and tasks, and for studying their computational and mathematical properties. We also believe it will be useful for developing an operational framework and system in which many existing neurosymbolic AI systems can be emulated, see (... forthcoming).

References

[1] Kareem Ahmed, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck, and Antonio Vergari. Semantic probabilistic layers for neuro-symbolic learning. *Advances in Neural Information Processing Systems*, 35:29944–29959, 2022.

[2] Eriq Augustine, Connor Pryor, Charles Dickens, Jay Pujara, William 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.

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

[4] Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artificial Intelligence*, 303:103649, 2022.

[5] Vaishak Belle, Andrea Passerini, Guy Van den Broeck, et al. Probabilistic inference in hybrid domains by weighted model integration. In *Proceedings of 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2770–2776. AAAI Press/International Joint Conferences on Artificial Intelligence, 2015.

[6] 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. *CoRR*, abs/1711.03902, 2017.

[7] Mark Chavira and Adnan Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6-7):772–799, 2008.

[8] Tirtharaj Dash, Sharad Chitlangia, Aditya Ahuja, and Ashwin Srinivasan. How to tell deep neural networks what we know. *arXiv preprint arXiv:2107.10295*, 2021.

[9] Artur S. 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. *FLAP*, 6, 2019.

[10] Luc De Raedt, Kristian Kersting, Sriraam Natarajan, and David Poole. *Statistical Relational Artificial Intelligence: Logic, Probability, and Computation*. Morgan & Claypool Publishers, 2016. doi: 10.2200/S00692ED1V01Y201601AIM032. URL `https://doi.org/10.2200/S00692ED1V01Y201601AIM032`.

[11] Luc De Raedt, Sebastijan Dumančić, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International*

Joint Conference on Artificial Intelligence, IJCAI-20, pages 4943–4950. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/688. URL `https://doi.org/10.24963/ijcai.2020/688`. Survey track.

[12] Lauren Nicole DeLong, Ramon Fernández Mir, Zonglin Ji, Fiona Niamh Coulter Smith, and Jacques D Fleuriot. Neurosymbolic ai for reasoning on biomedical knowledge graphs. *arXiv preprint arXiv:2307.08411*, 2023.

[13] Artur d'Avila Garcez and Luís C. Lamb. Neurosymbolic AI: The 3rd Wave. *Artificial Intelligence Review*, 2023. doi: 10.1007/s10462-023-10448-w. URL `https://doi.org/10.1007/s10462-023-10448-w`.

[14] Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Luis C Lamb, Leo de Penning, BV Illuminoo, Hoifung Poon, and COPPE Gerson Zaverucha. Neural-symbolic learning and reasoning: a survey and interpretation. *Neuro-Symbolic Artificial Intelligence: The State of the Art*, 2022.

[15] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In *ICLP/SLP*, volume 88, pages 1070–1080. Cambridge, MA, 1988.

[16] Madan M Gupta and J11043360726 Qi. Theory of t-norms and fuzzy inference methods. *Fuzzy sets and systems*, 40(3):431–450, 1991.

[17] Pascal Hitzler and Md. Kamruzzaman Sarker, editors. *Neuro-Symbolic Artificial Intelligence: The State of the Art*, volume 342 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2021. ISBN 978-1-64368-244-0. doi: 10.3233/FAIA342. URL `https://doi.org/10.3233/FAIA342`.

[18] Sepp Hochreiter. Toward a broad AI. *Communications of the ACM*, 65(4): 56–57, 2022.

[19] Ying Jiao, Luc De Raedt, and Giuseppe Marra. Valid text-to-sql generation with unification-based deepstochlog. In *International Conference on Neural-Symbolic Learning and Reasoning*, pages 312–330. Springer, 2024.

[20] Henry A. Kautz. The third AI summer: AAAI Robert S. Engelmore memorial lecture. *AI Mag.*, 43(1):93–104, 2022. doi: 10.1609/aimag.v43i1.19122. URL `https://doi.org/10.1609/aimag.v43i1.19122`.

[21] Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. *advances in neural information processing systems*, 31, 2018.

[22] Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence*, 298:103504, September 2021. ISSN

00043702. doi: 10.1016/j.artint.2021.103504. URL https://linkinghub.elsevier.com/retrieve/pii/S0004370221000552.

[23] Giuseppe Marra and Ondřej Kuželka. Neural markov logic networks. In *Uncertainty in Artificial Intelligence*, pages 908–917. PMLR, 2021.

[24] Giuseppe Marra, Sebastijan Dumančić, Robin Manhaeve, and Luc De Raedt. From statistical relational to neurosymbolic artificial intelligence: A survey. *Artificial Intelligence*, page 104062, 2024.

[25] Simon Odense and Artur d'Avila Garcez. A semantic framework for neurosymbolic computation. *Artificial Intelligence*, 340:104273, 2025.

[26] Connor Pryor, Charles Dickens, Eriq Augustine, Alon Albalak, William Wang, and Lise Getoor. Neupsl: Neural probabilistic soft logic. *arXiv preprint arXiv:2205.14268*, 2022.

[27] Enrique H Ruspini. On the semantics of fuzzy logic. *International Journal of Approximate Reasoning*, 5(1):45–88, 1991.

[28] Xiaoting Shao, Alejandro Molina, Antonio Vergari, Karl Stelzner, Robert Peharz, Thomas Liebig, and Kristian Kersting. Conditional sum-product networks: Imposing structure on deep probabilistic architectures. In *International Conference on Probabilistic Graphical Models*, pages 401–412. PMLR, 2020.

[29] Xiaowu Sun and Yasser Shoukry. Neurosymbolic motion and task planning for linear temporal logic tasks. *IEEE Transactions on Robotics*, 2024.

[30] Michael van Bekkum, Maaike de Boer, Frank van Harmelen, André Meyer-Vitali, and Annette ten Teije. Modular Design Patterns for Hybrid Learning and Reasoning Systems: a taxonomy, patterns and use cases. *arXiv:2102.11965 [cs]*, March 2021. URL http://arxiv.org/abs/2102.11965. arXiv: 2102.11965.

[31] Allen Van Gelder, Kenneth A Ross, and John S Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM (JACM)*, 38(3):619–649, 1991.

[32] Frank van Harmelen and Annette ten Teije. A boxology of design patterns for hybrid learning and reasoning systems. *J. Web Eng.*, 18(1-3):97–124, 2019. doi: 10.13052/jwe1540-9589.18133. URL https://doi.org/10.13052/jwe1540-9589.18133.

[33] Emile van Krieken, Samy Badreddine, Robin Manhaeve, and Eleonora Giunchiglia. Uller: A unified language for learning and reasoning. In *International Conference on Neural-Symbolic Learning and Reasoning*, pages 219–239. Springer, 2024.

[34] Thomas Winters, Giuseppe Marra, Robin Manhaeve, and Luc De Raedt. DeepStochLog: neural stochastic logic programming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10090–10100, 2022.

[35] Wen-Chi Yang, Giuseppe Marra, Gavin Rens, and Luc De Raedt. Safe reinforcement learning via probabilistic logic shields. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 5739–5749, 2023.

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

[37] Lotfi A Zadeh. Fuzzy sets. *Information and Control*, 1965.

Appendix A. Basic of measure theory

Luc: do we really need this appendix – seems to add nothing except the definition of meausurable spaces ? Lennert: I am not sure, these are really the absolute basics of measure theory so I guess we can also just rely on this being "common" knowledge.

Our definition of neurosymbolic inference uses certain basic concepts from measure theory that we outline here.

Definition Appendix A.1 (σ-algebra and measurable spaces)**.** Let X be a set, then a *σ-algebra* Σ on X is a non-empty collection of subsets of X that satisfies the properties

1. $\forall S \in \Sigma : S^c \in \Sigma$,
2. $\forall (S_n)_{n\in\mathbb{N}} : (\forall i \in \mathbb{N} : S_i \in \Sigma) \implies \bigcup_{n\in\mathbb{N}} S_n \in \Sigma$,
3. $\forall (S_n)_{n\in\mathbb{N}} : (\forall i \in \mathbb{N} : S_i \in \Sigma) \implies \bigcap_{n\in\mathbb{N}} S_n \in \Sigma$.

That is, a σ-algebra Σ is closed with respect to taking the complement, countable unions and countable intersections. If Σ is a σ-algebra on the set X, then the couple (X, Σ) is called a *measurable space*. A function f between two measurable spaces (S, Σ_S) and (T, Σ_T) is called *measurable* if $f^{-1}(T) \in \Sigma_S$ for each $T \in \Sigma_T$.

Example Appendix A.2 (A σ-algebra for the Boolean interpretations of propositional logic)**.** Assume we limit the set A of atomic expressions of the language of propositional logic to be finite, e.g. the modern Latin alphabet. In this case, the set of all possible Boolean interpretations is isomorphic to $\mathbb{B}^{26}$. Any finite set can easily be provided with a σ-algebra by taking the powerset of that set, so a σ-algebra of the set $\mathbb{B}^{26}$ of interpretations could be $\mathcal{P}(\mathbb{B}^{26})$. It is trivial to verify that this collection indeed satisfies the necessary conditions to be a σ-algebra of $\mathbb{B}^{26}$.

Definition Appendix A.3 (Measure)**.** Let (X, Σ) be a measurable space, then a function $\sigma : \Sigma \to \mathbb{R} \cup \{\infty\}$ is called a *measure* if it satisfies

1. $\sigma(\varnothing) = 0$,
2. **Non-negativity**: $\forall S \in \Sigma : \sigma(S) \geq 0$,
3. **Sigma-additivity**: $\forall (S_n)_{n\in\mathbb{N}} : (\forall i, j, l \in \mathbb{N} : S_i \in \Sigma \wedge S_j \cap S_l = \varnothing) \implies \sigma\left(\bigcup_{n\in\mathbb{N}} S_n\right) = \sum_{n\in\mathbb{N}} \sigma(S_n)$.

In other words, a measure is a positive map of subsets of X to the extended real number line that "commutes" with countable unions.

Example Appendix A.4 (A measure for the Boolean interpretations of propositional logic)**.** Assume the same setting as in Example Appendix A.2 and take the measurable space $(\Omega, \mathcal{P}(\Omega))$ with $\Omega = \mathbb{B}^{26}$. A well-known measure for finite measurable spaces is the *counting measure* σ_C that outputs the cardinality of each element of Σ, i.e. $\sigma_C(S) = |S|$.