On the Hardness of Probabilistic Neurosymbolic Learning

Jaron Maene [1] **Vincent Derkinderen** [1] **Luc De Raedt** [1 2]

Abstract

The limitations of purely neural learning have sparked an interest in probabilistic neurosymbolic models, which combine neural networks with probabilistic logical reasoning. As these neurosymbolic models are trained with gradient descent, we study the complexity of differentiating probabilistic reasoning. We prove that although approximating these gradients is intractable in general, it becomes tractable during training. Furthermore, we introduce *WeightME*, an unbiased gradient estimator based on model sampling. Under mild assumptions, WeightME approximates the gradient with probabilistic guarantees using a logarithmic number of calls to a SAT solver. Lastly, we evaluate the necessity of these guarantees on the gradient. Our experiments indicate that the existing biased approximations indeed struggle to optimize even when exact solving is still feasible.

1. Introduction

Neurosymbolic artificial intelligence aims to combine the strengths of neural and symbolic methods into a single unified framework (Garcez et al., 2019; Hitzler, 2022; Marra et al., 2024). One prominent strain of neurosymbolic models combines neural networks with probabilistic reasoning (Manhaeve et al., 2018; Xu et al., 2018; Yang et al., 2021; Ahmed et al., 2023). These models are state-of-the-art but suffer from scalability issues due to the #P-hard nature of probabilistic inference. Probabilistic inference has been extensively studied on probabilistic graphical models (Koller & Friedman, 2009) and model counting (Chakraborty et al., 2021). However, neurosymbolic learning deviates from regular probabilistic inference due to its learning aspect: the goal is to efficiently obtain useful gradients rather than to compute the exact probabilities. We therefore investigate the *approximation of gradients for probabilistic inference* in its own right and prove several positive and negative results.

In Section 3, we show that the intractability of probabilistic inference also implies that it is impossible to approximate gradients with probabilistic guarantees in polynomial time (unless RP = NP, where RP is randomized polynomial time). However, in Section 4, we prove that the gradient approximation problem *becomes tractable during training* when the neural network outputs converge to binary values. On the negative side, we find that this tractable region can become unreachable for large problems.

In Section 5, we investigate more general-purpose gradient estimators with strong guarantees by allowing access to a SAT oracle. We introduce the *Weighted Model Estimator (WeightME)*, a novel gradient estimator for probabilistic inference which relies on weighted model sampling. As opposed to existing approximations for neurosymbolic learning (Huang et al., 2021; Manhaeve et al., 2021; Ahmed et al., 2022; Li et al., 2022; van Krieken et al., 2023; Verreet et al., 2023), WeightME is unbiased and probably approximately correct under mild assumptions. Furthermore, WeightME only needs a logarithmic number of SAT invocations in the number of variables.

Finally, Sections 6 and 7 provide a comprehensive overview and evaluation of existing approximation techniques. Our results indicate that they have difficulties optimizing benchmarks that can still easily be solved exactly. This suggests that principled methods are warranted if we want to apply probabilistic neurosymbolic optimization to more complex reasoning tasks.

2. Weighted Model Counting

Probabilistic inference can be reduced to *weighted model counting* (WMC). WMC has been called "an assembly language for probabilistic reasoning" (Belle et al., 2015) as a wide range of probabilistic models can be cast into WMC (Chavira & Darwiche, 2008; Domshlak & Hoffmann, 2007; Suciu et al., 2011; Holtzen et al., 2020; Derkinderen et al., 2024). Bayesian networks are a notable example where WMC solvers are state-of-the-art for exact inference (Chavira & Darwiche, 2008). Importantly, WMC also

[1]KU Leuven, Department of Computer Science, Leuven, Belgium. [2]Örebro University, Centre for Applied Autonomous Sensor Systems, Örebro, Sweden. Correspondence to: Jaron Maene <jaron.maene@kuleuven.be>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

underlies probabilistic inference in NeSy frameworks (Manhaeve et al., 2018; Xu et al., 2018; Huang et al., 2021; Ahmed et al., 2022).

We briefly introduce propositional logic and WMC. Propositional *variables* are denoted with lowercase letters (e.g. x or y). A *literal* is a variable x or its negation $\neg x$. A propositional *formula* ϕ combines logical variables with the usual connectives: negation $\neg\phi$, conjunction $\phi_1 \wedge \phi_2$, and disjunction $\phi_1 \vee \phi_2$. A *clause* is a disjunction of literals. A *CNF* formula is a conjunction of clauses. A *DNF* formula is a disjunction of conjunctions of literals. We write $\text{var}(\phi)$ for the number of variables in a formula ϕ. We use $\phi \mid x$ to denote the formula ϕ conditioned on x being true, i.e. every occurrence of x (resp. $\neg x$) in ϕ is replaced by true (resp. false).

An *interpretation* I is a set of literals representing an instantiation of the variables. We say that a variable x is true (resp. false) in the interpretation I when $x \in I$ (resp. $\neg x \in I$). An interpretation I is a *model* of the formula ϕ, denoted as $I \models \phi$, when ϕ is satisfied under the truth assignment of I. The *satisfiability problem (SAT)* asks whether a formula has at least one model, while *model counting (#SAT)* asks how many models a formula has. The *weighted model count* (WMC) is a weighted sum over the models.

Definition 2.1 (Weighted Model Count)**.** Given a propositional logic formula ϕ and a weight function w that maps every literal to a real number, the weighted model count is

$$\text{WMC}(\phi; w) = \sum_{I : I \models \phi} \prod_{l \in I} w(l) \tag{1}$$

Both #SAT and WMC are #P-complete problems (Valiant, 1979), and even approximating them with probabilistic guarantees is NP-hard (Roth, 1996).

Due to our probabilistic focus, we only consider weights corresponding to Bernoulli distributions. More concretely, we assume $w(x) \in [0, 1]$ and $w(x) = 1 - w(\neg x)$ for every variable x. Consequently, we have a probability distribution over interpretations: $P(I; w) = \prod_{l \in I} w(l)$.

Example 1. *Consider the formula* $\phi = (a \vee b) \wedge (\neg b \vee c)$ *with weights* $w(a) = 0.5$*,* $w(b) = 0.1$*, and* $w(c) = 0.25$*. This formula has four models:* $\{a, \neg b, \neg c\}$*,* $\{\neg a, b, c\}$*,* $\{a, \neg b, c\}$*, and* $\{a, b, c\}$*. When we sum the probabilities of these models we get the weighted model count.*

$$\textit{WMC}(\phi, w) = 0.3375 + 0.0125 + 0.1125 + 0.0125 = 0.475$$

We omit the weight function w when it is clear from context. In Appendix A, we further discuss how to handle categorical distributions and unweighted variables in WMC and explain how inference on a Bayesian network reduces to WMC.

In the neurosymbolic context, the weights w are the probabilities produced by a neural network. Typically, these weights will be random at initialization and get closer to zero or one during training as the neural network becomes more confident in its predictions.

3. From WMC to ∇WMC

Probabilistic neurosymbolic methods optimize the WMC by iteratively updating the weights w with gradient descent. So just like inference in probabilistic models can be reduced to WMC, learning in probabilistic neurosymbolic models can be reduced to taking the gradient of the WMC.

$$\nabla_w \text{WMC}(\phi; w) = \left[\frac{\partial \text{WMC}(\phi; w)}{\partial w(x_1)}, \ldots, \frac{\partial \text{WMC}(\phi; w)}{\partial w(x_{\text{var}(\phi)})} \right]^\top$$

This WMC gradient is the core focus of our work, and as the next theorem shows, is closely related to the WMC itself.

Theorem 3.1. *Computing the partial derivative of a WMC problem is reducible to WMC problems, and vice versa.*

Proof. Both directions follow from the decomposition

$$\text{WMC}(\phi) = w(x)\text{WMC}(\phi \mid x) + w(\neg x)\text{WMC}(\phi \mid \neg x)$$

The insight here is that $\text{WMC}(\phi \mid x)$ and $\text{WMC}(\phi \mid \neg x)$ do not have a gradient for $w(x)$, implying that

$$\frac{\partial \text{WMC}(\phi, w)}{\partial w(x)} = \text{WMC}(\phi \mid x) - \text{WMC}(\phi \mid \neg x)$$

For the other direction, we introduce a dummy variable t.

$$\text{WMC}(\phi, w) = \frac{\partial \text{WMC}(\phi \wedge t, w)}{\partial w(t)} \qquad \square$$

Theorem 3.1 allows us to study the ∇WMC problem using existing results on WMC. Notably, it follows immediately that computing the exact gradients is #P-complete, just like for WMC. Gradient descent does not necessarily need exact gradients, however. We therefore investigate approximations of ∇WMC. In particular, we focus on approximations that 1) are unbiased and 2) have probabilistic guarantees. Informally, this means that the approximation 1) is correct in expectation and 2) has a high probability of being close to the true gradient. This probabilistic guarantee is formalized as an (ϵ, δ)-approximation.

Definition 3.2. An estimator $\hat{y}$ is an (ϵ, δ)-approximation for y when $P\left(|(y - \hat{y})/y| > \epsilon\right) \leq \delta$.

The (ϵ, δ)-approximation enforces a probabilistic bound on the relative error: the probability of having a relative error larger than ϵ is at most δ. This guarantee is also known as probably approximately correct (PAC). Unfortunately, (ϵ, δ)-approximations of the WMC are still hard, and due to Theorem 3.1 the same can be said for derivatives.

Theorem 3.3. *Computing an (ϵ, δ)-approximation of the partial derivative $\partial WMC(\phi)/\partial w(x)$ is NP-hard.*

Proof. Follows from Theorem 3.2 of Roth (1996) and Theorem 3.1. □

It is worth reflecting on why an (ϵ, δ)-approximation of the gradient is relevant as opposed to the variance of the gradient, which is more commonly discussed. The variance $\text{Var}[\hat{y}] = \mathbb{E}[(y - \hat{y})^2]$ measures the squared error instead of the relative error. However, a low squared error is not meaningful when the WMC and ∇WMC are near zero. In Appendix B, we demonstrate how existing gradient estimators for ∇WMC achieve low variance by simply returning zero gradients. An (ϵ, δ)-approximation does not suffer from this problem as it measures the relative instead of the squared error.

4. The (In)tractability of Sampling

One way to get an unbiased (ϵ, δ)-approximation for the WMC is by sampling. Indeed, the WMC can be seen as the expectation that a random interpretation is a model.

$$\text{WMC}(\phi; w) = \mathbb{E}_{I \sim P(I;w)}[\mathbb{1}(I \models \phi)] \tag{2}$$

Here, $\mathbb{1}(\cdot)$ denotes the indicator function. Equation 2 leads to a straightforward Monte Carlo approximation, which we call *interpretation sampling*. To the best of our knowledge, all existing unbiased estimators for ∇WMC are based on interpretation sampling. We can obtain gradients with conventional techniques such as the score function estimator (SFE), also known as REINFORCE (Sutton et al., 1999).

$$\nabla_w \text{WMC}(\phi; w) = \mathbb{E}_{I \sim P(I;w)}\, \mathbb{1}(I \models \phi)\, \nabla_w \log P(I; w)$$

Alternatively, we can use the decomposition of Theorem 3.1. This corresponds to the IndeCateR estimator, which is a Rao-Blackwellization of the SFE (De Smet et al., 2023).

$$\frac{\partial \text{WMC}(\phi, w)}{\partial w(x)} = \mathbb{E}_{I \sim P(I|x;w)}\, \mathbb{1}(I \models \phi) - \mathbb{E}_{I \sim P(I|\neg x;w)}\, \mathbb{1}(I \models \phi)$$

From Theorem 3.3, we already know that interpretation sampling cannot give a (ϵ, δ)-approximation in polynomial time. Indeed, Karp et al. (1989) showed that $c(\epsilon, \delta)/\text{WMC}(\phi)$ samples are required for an (ϵ, δ)-approximation of $\text{WMC}(\phi)$, where $c(\epsilon, \delta)$ is in the order of $\epsilon^{-2} \log \frac{2}{\delta}$. The problematic part here is $1/\text{WMC}(\phi)$ because the WMC can decrease exponentially in the number of variables[1].

[1]To see this, consider the case where ϕ has a single model and all weights are $w(x) = \frac{1}{2}$, so that $\text{WMC}(\phi) = 2^{-var(\phi)}$.

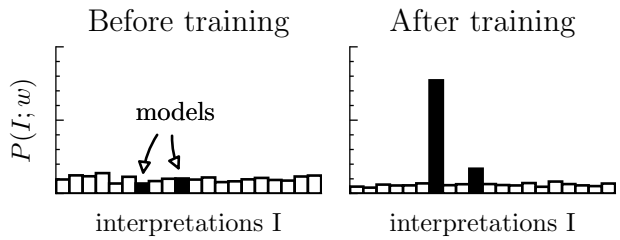


Figure 1: When sampling from the distribution of interpretations, we need to hit a model to obtain a gradient. (left): At initialization, the distribution over interpretations is fairly uniform and the probability that interpretation sampling finds a model is vanishingly small. Model sampling avoids this by sampling directly from the models. (right): When the neural network becomes more confident in its predictions, it becomes easier to sample a model.

4.1. Example of Tractability

In the neurosymbolic setting, interpretation sampling can *become tractable during training*. This occurs when the neural network becomes more confident, i.e. when the weights $w(x)$ approach zero or one. We illustrate this phenomenon on the pedagogical MNIST-addition task.

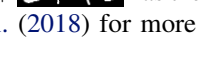
In the MNIST-addition task, a digit classifier is trained with distant supervision. There are no labels for the individual MNIST images, but only on the sums of two numbers represented by images. For example, the input  has the label 3916. We refer to Manhaeve et al. (2018) for more details on the experimental setup.

Figure 2 displays the first epoch of training with exact inference on MNIST-addition with 4 digits. At every iteration, we estimate the gradient with the SFE and compare it with the true gradient. Initially, the neural network is random, and estimating the gradients is infeasible. But at about 700 training iterations, there is a transition after which the sampled gradient becomes a faithful approximation.

4.2. Tractability from Training

We formalize the above example using implicants. An *implicant* of a formula ϕ is a conjunction of literals that models a subset of the models of ϕ. Now, we prove that an (ϵ, δ)-approximation of the gradient becomes tractable when a single implicant dominates the WMC.

Theorem 4.1. *The partial derivative of WMC(ϕ) with respect to $w(x)$ admits a polynomial time (ϵ, δ)-approximation when there exists a implicant π such that $x \in \pi$ and WMC$(\pi) \geq$ WMC$(\phi \mid \neg x) + c(\epsilon, \delta)$.*

Proof. See Appendix C. □

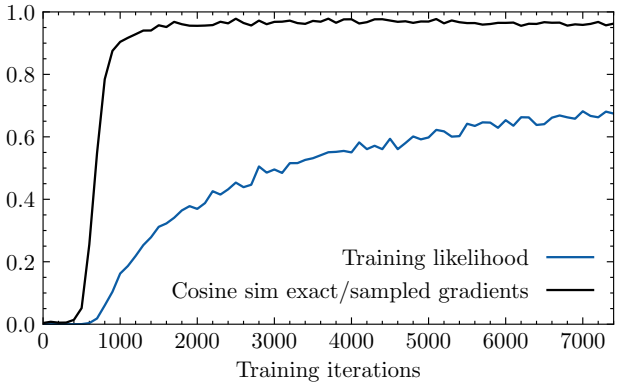


Figure 2: First epoch of training on 4-digit MNIST-addition with exact inference. We also plot the cosine similarity between the exact gradients and the sampled gradients from the SFE. Results are averaged over 10 seeds.

The assumption that an implicant dominates the probability mass towards the end of training has been observed before empirically (Manhaeve et al., 2021) and was recently proven to hold for all minima of a WMC problem (van Krieken et al., 2024).

As our definition of an (ϵ, δ)-approximation pertains to a single partial derivative, some partial derivatives in the gradient may be tractable while others are not. For Theorem 4.1 to apply, we need $\text{WMC}(\phi \mid \neg x)$ to converge to zero as $\text{WMC}(\pi)$ converges to one. This will generally be the case, unless a different implicant π' partly covers both π and $(\pi \mid x) \wedge \neg x$. Or more informally, when tractability is not achieved at convergence, it is precisely because the value of this literal does not matter for the WMC. For example, this can occur because π is not prime, meaning $\pi \mid x$ is still an implicant.

4.3. Tractability from Concept Supervision

The practical relevance of Theorem 4.1 depends on how close the neural network must be to convergence before we end up in the tractable region. We formalize this as follows.

Definition 4.2. Consider a formula ϕ with weights w and M the most probable model. We say that ϕ is *τ-supervised* when $\tau = \#\{x \mid x \in M \text{ and } w(x) > 1/2\}$.

In other words, the weights are τ-supervised when the neural network classifies τ of the $\text{var}(\phi)$ weights confidently with respect to a model M.

Theorem 4.3. *To ensure a polynomial time (ϵ, δ)-approximation of the partial derivative $\partial \text{WMC}(\phi)/\partial w(x)$ for a τ-supervised formula ϕ with interpretation sampling, it is required that $var(\phi) - c'(\epsilon, \delta) \leq \tau$.*

Proof. See Appendix C. □

Theorem 4.3 says that in the worst case, the number of variables that should be τ-supervised is close to the total number of variables. More precisely, the percentage of variables that are allowed to be misclassified by the neural network decreases as $1/var(\phi)$. Firstly, this implies that the tractability of Theorem 4.1 might not always be reachable in practice. Secondly, Theorem 4.3 suggests that concept supervision cannot entirely alleviate the need for approximate inference. Concept supervision uses some direct supervision on the weights of logical variables, instead of training from scratch. For example, in the MNIST-addition example, some images get an individual label (e.g. $= 9$).

5. Weighted Model Sampling

When the WMC is too small for interpretation sampling, we need to look elsewhere to obtain gradients. The past decades have seen considerable progress in approximate model counting, despite its NP-hardness (Chakraborty et al., 2021). Leveraging this progress for ∇WMC could push the scalability of gradient approximations with guarantees.

Stockmeyer (1983) proved that model counting with PAC guarantees is possible using a polynomial number of SAT calls. More recently, Chakraborty et al. (2016) sharpened this to a logarithmic number of SAT calls in the number of variables. The state-of-the-art in approximate unweighted model counting implements this with hash-based methods (Soos & Meel, 2019). In short, they use hash functions to randomly partition the space of interpretations and count models within these partitions.

Hash-based methods could be applied immediately to obtain gradients using the decomposition of Theorem 3.1. However, that would require $2\text{var}(\phi)$ calls to the approximate counter to calculate a single gradient, as every partial derivative is computed separately. Furthermore, the subtraction of two (ϵ, δ)-approximations weakens the provided guarantees.

We can do better by instead relying on model sampling. *Weighted model sampling* (WMS) is the task of sampling models from a weighted formula ϕ such that the probability of selecting a model M is $P(M)/\text{WMC}(\phi)$. Using WMS, we introduce the following estimator.

Definition 5.1. The *weighted model estimator* (WeightME) is defined as

$$\frac{\partial \log \text{WMC}(\phi)}{\partial w(x)} = \mathbb{E}_M \left[\frac{\mathbb{1}(x \in M)}{w(x)} - \frac{\mathbb{1}(x \notin M)}{w(\neg x)} \right]$$

With $\mathbb{E}_M$ we denote an expectation over the models of ϕ. WeightME is unbiased and achieves probabilistic guarantees using only a constant number of samples.

Theorem 5.2. *WeightME is an unbiased estimator for $\partial \log \text{WMC}(\phi)/\partial w(x)$ when $w(x) \in (0, 1)$.*

Proof. See Appendix C. □

Theorem 5.3. *Given WMC(ϕ), WeightME can (ϵ, δ)-approximate the partial derivative ∂WMC(ϕ)/$\partial w(x)$ using a constant number of weighted model samples when there is a constant $\lambda > 0$ such that $|P(x \mid \phi) - w(x)| > \lambda$ and $w(x) \in (0, 1)$.*

Proof. The proof relies on the observation that

$$\frac{\partial \text{WMC}(\phi)}{\partial w(x)} = \frac{\text{WMC}(\phi)}{w(\neg x)} \cdot \left(\frac{\mathbb{E}_M[\mathbb{1}(x \in M)]}{w(x)} - 1 \right)$$

With $c(\epsilon, \delta)/\lambda^2$ weighted model samples, the above is an (ϵ, δ)-approximation. The full proof is in Appendix C. □

This last theorem has several implications.

1. The required number of model samples for an (ϵ, δ)-approximation does not increase with formula size. Indeed, the condition that $w(x)$ and WMC(ϕ) are dependent does not rely on the formula size. So in contrast to Theorem 4.1, Theorem 5.3 scales to large formulas where WMC(ϕ) is very small.
2. Theorems 4.1 and 5.3 can be seen as complementary. The variance of IndeCateR is zero when the weights are binary, while WeightME has a low variance when the weights are close to 1/2.
3. The condition in Theorem 5.3 of having WMC(ϕ) can be dropped if we are satisfied with $\nabla \log$ WMC instead of ∇WMC. This is a reasonable assumption as the WMC tends to be optimized with a negative log-likelihood loss.

5.1. Approximate model sampling

The question remains how to actually sample the models. WMS and WMC are polynomially inter-reducible, so exact WMS is also #P-hard (Jerrum et al., 1986). Scalable WMS methods hence resort to approximations and do not sample exactly according to the weighted model distribution $P(M)/\text{WMC}(\phi)$. Crucially, Theorem 5.3 can still apply when the WMS samples are (ϵ, δ)-approximate (see Appendix C). A single WeightME gradient is therefore possible with only a logarithmic number of calls to a SAT oracle (Chakraborty et al., 2016).

Just as for WMC, approximate WMS with PAC guarantees is implemented with hash-based techniques (Soos et al., 2020). The unweighted variant of model sampling has received more attention than WMS (Chakraborty et al., 2014). However, it is possible to convert weighted into unweighted problems (Chakraborty et al., 2015), which could make it possible to leverage the progress in unweighted sampling for gradient approximation.

Biased WMS approximations have the promise to scale further, but lack guarantees. Golia et al. (2021) investigated this trade-off, by using sampling testers on biased approximations. This led them to propose a performant solver that samples from a distribution that does not differ from the true distribution on statistical tests. Markov-Chain Monte-Carlo (MCMC) techniques have also been proposed for model sampling (Ermon et al., 2012). Unfortunately, combinatorial problems face an exponential mixing time of the Markov chains. Li et al. (2022) addressed this problem by using projection techniques from SMT solvers.

A promising recent development is the sampling of models using neural approximations. van Krieken et al. (2023) connected the theory of GFlowNets (Bengio et al., 2023) to model sampling. However, to the best of our knowledge, this has not yet been realized on practical WMC problems.

6. Biased WMC

Various approximate inference methods exist that trade in guarantees for better scalability, and are often not NP-hard. Theory does not always align with practice, and it is hence not implausible that some of these could be competitive.

We give a comprehensive overview of relevant approximations for WMC but do not aim to be exhaustive. Instead, we focus on the most notable and common approaches. We summarize all methods in Table 1.

6.1. Biased inference

The following methods compute a biased yet differentiable approximation for the WMC.

k-Best approximates a formula using a DNF, containing the k implicants with the highest probability (Kimmig et al., 2008; Manhaeve et al., 2021; Huang et al., 2021). This works well when the probability mass of a formula can be captured by only a small number of implicants, for which exact WMC is then feasible. For $k = 1$, k-best coincides with the most probable explanation (MPE).

The probability mass of the k-best implicants can overlap heavily, so **k-optimal** greedily searches for the k-DNF with the maximum total probability (Renkens et al., 2012). Finding these implicants reduces to iteratively solving weighted MaxSAT problems (Renkens et al., 2014).

Uniform model sampling samples models uniformly without considering the weights. As with k-best, these models can be used as a lower bound for the true WMC. Verreet et al. (2023) argue for this approach to maximize the diversity of samples during training.

Fuzzy t-norms are arguably the most common neurosymbolic semantics. They replace the logical (Boolean) oper-

Table 1: Classification of all considered approximation methods according to whether they are unbiased, deterministic, are learned from data, what formulas they support, and their complexity. For the complexity, we denote the number of clauses as c, and the number of samples as s. We assume the clause length is bounded, and the number of variables is linear in the size of the clause.

Method	Unbiased	Deterministic	Learned	Formula	Time Complexity
Interpretation sampling	Y	N	N	Any	cs
Weighted model sampling	Y	N	N	Any	NP-hard
Hash-based	Y	N	N	Any	NP-hard
Fuzzy t-norms	N	Y	N	Any	c
MPE / k-best	N	Y	N	Any	NP-hard
Unweighted model sampling	N	N	N	Any	NP-hard
Neural#DNF	N	Y	Y	DNF	c
A-NeSI	N	Y	Y	Any	c
Collapsed sampling	Y	N	N	Any	cs
Bounded inference	/	Y	N	Any	c
Semantic strengthening	N	Y	N	Any	c^2
Straight-trough estimator	N	N	N	Any	cs
Gumbel-Softmax	N	N	N	Any	cs
I-MLE	N	N	N	Any	NP-hard

ations with continuous generalizations (Badreddine et al., 2022; van Krieken et al., 2022). For example, the product t-norm computes conjunction as $w(x \wedge y) = w(x) \cdot w(y)$ and disjunction as $w(x \vee y) = 1 - w(\neg x) \cdot w(\neg y)$. Crucially, fuzzy semantics has linear complexity in the size of the propositional formula. T-norms can also be used as an approximation of probabilistic semantics. For example, the product t-norm computes the WMC of a CNF under the assumption that all clauses are independent.

Neural approximations. As neural networks are universal approximators, recent works have proposed using them to approximate probabilistic inference. From the algebraic view, Zuidberg Dos Martires (2021) introduces the neural semiring, where a neural network learns the algebraic operations for conjunction and disjunction. Abboud et al. (2020) realize this with graph neural networks. The authors limit the scope to DNFs, motivated by the tractability of this special case.

6.2. Biased gradient estimation

Several gradient estimators have been proposed to differentiate through sampling and hence can be applied to Equation 2. Notably, several biased estimators can still give a gradient when the WMC is tiny.

When estimating the gradients of an expectation of the form $\mathbb{E}_{x \sim P(x)} f(x)$, biased estimators typically assume that f is continuous. As such, we need a continuous relaxation for $\mathbb{1}(I \models \phi)$. The straightforward solution we consider is to use fuzzy semantics to determine whether an interpretation is a model of the formula.

The **straight-through estimator** (STE) replaces the sample with its probability during backpropagation (Bengio et al., 2013). **Gumbel-Softmax** estimates gradients by replacing the Categorical distribution with the differentiable Gumbel-Softmax distribution (Jang et al., 2016; Maddison et al., 2016).

Implicit Maximum Likelihood Estimation (I-MLE) is a biased gradient estimator designed for combinatorial problems (Niepert et al., 2021). I-MLE approximates the gradient using a perturb-and-MAP approach: the weights are perturbed by adding some noise from a distribution such as Gumbel, and on these weights the most probable model is computed. I-MLE can hence be seen as an alternative to k-best rooted in perturbation-based implicit differentation.

6.3. Hybrid methods

Many of the approximation methods can be combined with each other, or with exact solving. We highlight some examples of this.

Collapsed sampling uses an exact solver up to a time or memory limit, after which the remaining components are estimated using interpretation sampling. Similar to exact solving, the choice of variable ordering greatly impacts the effectiveness of this method (Friedman & Van den Broeck, 2018).

Semantic strengthening combines exact solving with fuzzy t-norms (Ahmed et al., 2022). It uses mutual information

to determine which clause pairs violate the independence assumption most, and hence benefit from exact compilation. The remaining conjunctions are computed with a t-norm.

7. Experiments

It is clear that many methods exist to approximate WMC gradients, so the question arises as to which of the methods are appropriate in practice. For this reason, we evaluate the gradients of the various methods on a set of challenging WMC benchmarks.[2]

Benchmarks The model counting competition (MCC) is an annual competition on (weighted) model counting (Fichte et al., 2021). We take the benchmarks of the last three competitions (2021, 2023, and 2023) and take the instances that are probabilistic and can be solved exactly by state-of-the-art solvers (Lagniez & Marquis, 2017; Golia et al., 2021). As an easier benchmark, we also include the logical formula from the ROAD-R dataset (Giunchiglia et al., 2023), which imposes constraints on the object detection of self-driving cars. All benchmarks are CNF formulas. The weights are initialized with a Gaussian distribution with a mean of 1/2.

Setup All methods were executed on the same machine with an Intel Xeon E5-2690 CPU and used PyTorch to compute the gradients. WeightME was implemented using CMSGen (Golia et al., 2021) for WMS, as it was found to be the most scalable WMS solver available. For MPE and k-optimal we used the EvalMaxSAT solver (Avellaneda, 2020). The SFE was implemented with the Reinforce-Leave-One-Out baseline (Kool et al., 2019). All approximate methods got a timeout of 5 minutes per gradient. The true gradients were computed as ground truths using the d4 knowledge compiler (Lagniez & Marquis, 2017), which did not get a timeout.

7.1. Gradients at initialization

In the first experiment, we empirically validate which approximation methods succeed at the gradient estimation task at initialization.

Quality To evaluate the quality of the gradients, we compute the cosine similarity between the exact and approximate gradients on our set of benchmarks. Table 2 summarizes the results. WeightME attains the best results, both compared to polynomial and NP-hard methods. For the polynomial methods, the product t-norm and Gumbel-Softmax perform best. The Gödel t-norm performs weak, as by design it gives zero gradients to all but one variable. The SFE usually fails to sample a model, which is why it performs very poorly on these benchmarks. IndeCateR is not included in the results, as it suffers even harder from this problem. Unweighted model sampling performs similarly to WeightME on the smaller ROAD-R benchmark, but falls behind on more challenging problems.

Scalability In Figure 3, we look at the runtime of the various approaches. None of the NP-hard approximation methods manages to solve all benchmarks within the time limit (5 minutes/instance). Surprisingly, none of the tested MaxSAT or approximate WMS solvers could fully match d4. The polynomial methods scale quite well as expected, except for semantic strengthening, which quickly becomes infeasible when the number of clauses is high.

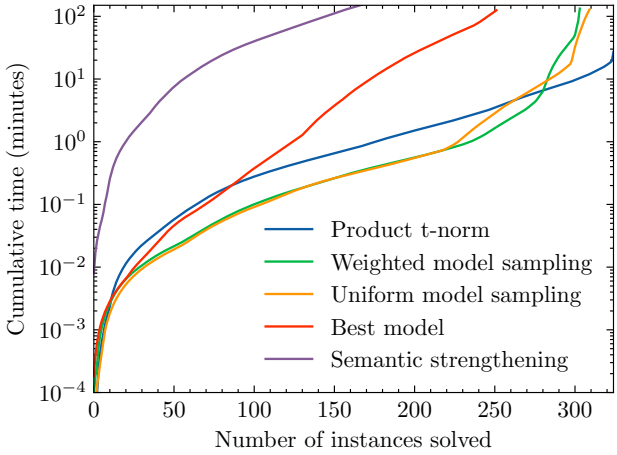


Figure 3: Cumulative runtimes on all the MCC instances. Omitted methods achieve similar performance to the Product t-norm.

7.2. Optimization

The previous experiments evaluate the quality of a single gradient in isolation. However, the real question is whether the approximate gradients suffice to optimize. The setup in Table 2 penalizes unbiased methods with high gradient variance. So in a second experiment, the task is to optimize the log-likelihood of the formulas. We can see this as a necessary (though not necessarily sufficient) requirement for probabilistic neurosymbolic learning. The NP-hard methods that explicitly find a model can trivially achieve this task and are omitted from this experiment.

In Figure 4, we let polynomial approximation methods optimize the log-likelihood of a formula on an easier subset of the MCC benchmarks. These benchmarks have fewer than 1000 variables and are well below the limit of state-of-the-art exact solvers. We plot the best achieved loss, within a maximum of 10000 iterations. Our results indicate that none of the tested approaches can consistently optimize. The re-

[2]The code to replicate these experiments can be found at https://github.com/jjcmoon/hardness-nesy.

Table 2: Benchmarks for gradient estimation at initialization. For each solver-benchmark combination, we list the average cosine similarity between the approximate and true gradient, as well as the standard deviation. Higher is better. We use − to indicate that at least one instance timed out.

	MCC2021	MCC2022	MCC2023	ROAD-R
Number of Instances	120	93	61	100
WeightME (k=100)	**0.784 ± 0.248**	**0.721 ± 0.259**	**0.821 ± 0.172**	**0.955 ± 0.009**
Unweighted model sampling (k=100)	-	0.578 ± 0.332	-	0.946 ± 0.014
MPE	-	-	-	0.609 ± 0.042
k-Optimal (k=100)	-	-	-	0.691 ± 0.034
Product t-norm	0.643 ± 0.259	0.592 ± 0.188	0.537 ± 0.310	0.918 ± 0.005
Gödel t-norm	0.092 ± 0.153	0.107 ± 0.119	0.103 ± 0.136	0.191 ± 0.094
Straight-through estimator (s=10)	0.195 ± 0.217	0.167 ± 0.202	0.309 ± 0.187	0.537 ± 0.078
Gumbel-Softmax estimator (s=10, τ=2)	0.584 ± 0.263	0.501 ± 0.187	0.510 ± 0.293	0.897 ± 0.020
SFE (s=10k)	0.035 ± 0.173	0.010 ± 0.093	0.000 ± 0.000	0.006 ± 0.064
Semantic strengthening (κ=100)	-	-	-	0.895 ± 0.019

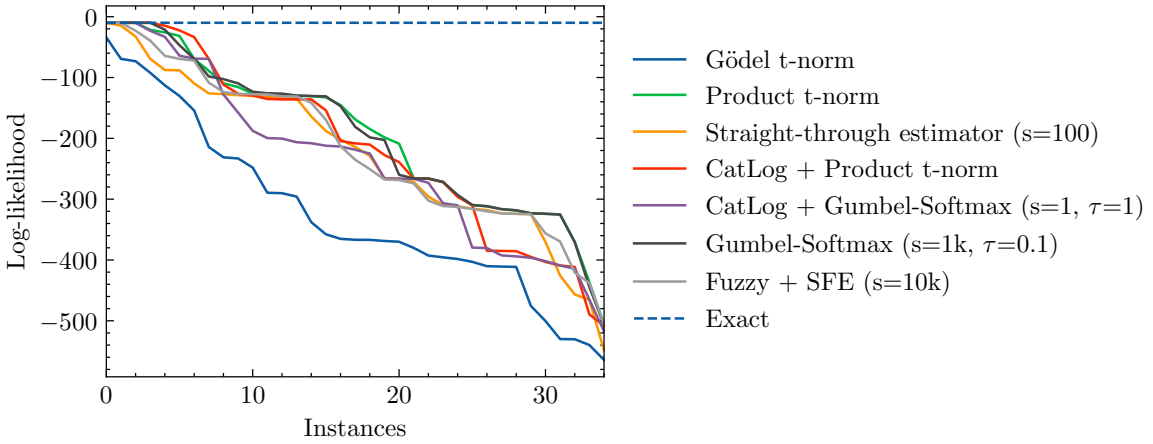


Figure 4: Maximum log-likelihood achieved by the various biased gradient approximations, sorted from best to worst. The benchmarks are 33 easy instances from the Model Counting Competitions. Higher is better.

sults in Figure 4 also include some novel baselines which are described in Appendix D. In Appendix E, we further validate that adding concept supervision does not alleviate the optimization problem.

8. Related work

The estimation of gradients is a well-studied problem in machine learning (Mohamed et al., 2020). Existing works have studied the gradient estimation of categorical distributions (Jang et al., 2016; Maddison et al., 2016; De Smet et al., 2023), while we are the first to focus on the gradients of WMC. van Krieken et al. (2022) analyze the gradients of fuzzy semantics in neurosymbolic learning. On the other hand, we target neurosymbolic learning with probabilistic semantics.

Related to our work, Niepert et al. (2021) propose a gradient estimator for black-box combinatorial solvers relying on the most probable model, which is both biased and harder to compute than approximate weighted model samples. Verreet et al. (2023) introduce a neurosymbolic optimization method using unweighted model samples, motivated by increasing the sample diversity. However, unweighted sampling provides weaker guarantees while being just as hard as weighted sampling.

Considerable progress has been made on approximating WMC and WMS (Wei & Selman, 2005; Gogate & Dechter, 2011; Ermon et al., 2012; Chakraborty et al., 2014; 2016; Soos & Meel, 2019; Golia et al., 2021; Soos et al., 2020). We focus on the gradients of WMC in a learning setting instead of approximating the WMC itself.

9. Limitations

As with any empirical study, the results of Section 7 are influenced by the choice of benchmarks, and might not generalize to all neurosymbolic tasks. Our work only considers propositional logic, while some neurosymbolic systems target the more expressive first-order logic. First-order neurosymbolic systems usually end up grounding their theory such that the propositional case studied here remains relevant. Inference for weighted first-order model counting is much harder still (Gribkoff et al., 2014), and hence the need to approximate is even more pertinent. We also do not consider DNFs, which in contrast to CNFs admit a tractable (ϵ, δ)-approximation (Karp et al., 1989).

10. Conclusion

We studied the gradient estimation of probabilistic reasoning by connecting this problem to weighted model counting. This allowed us to prove several results on the intractability, and prove how gradient estimation for neurosymbolic learning becomes tractable during training. Next, we contributed a general-purpose gradient estimator that builds on the progress in approximate counting and sampling. We showed that a constant number of weighted model samples is sufficient to achieve strong guarantees.

Finally, we turned our attention to existing approximation methods. Our experiments suggest that none of the polynomial methods can consistently optimize a formula. In contrast, existing NP-hard approximations typically struggle to scale, while lacking guarantees. Potential further work includes improving our understanding of the interaction between approximating the weighted model samples and the PAC guarantee of WeightME and expanding our analysis from propositional to first-order weighted model counting.

Acknowledgements

This research received funding from the Flemish Government (AI Research Program), the Flanders Research Foundation (FWO) under project G097720N, the KU Leuven Research Fund (C14/18/062) and TAILOR, a project from the EU Horizon 2020 research and innovation program under GA No 952215. Luc De Raedt is also supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

We are grateful to Lennert De Smet for his feedback on a draft of this paper, and to Pedro Zuidberg Dos Martires for a helpful discussion.

Impact Statement

This paper presents work whose goal is to advance the field of neurosymbolic artificial intelligence. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Abboud, R., Ceylan, I., and Lukasiewicz, T. Learning to Reason: Leveraging Neural Networks for Approximate DNF Counting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3097–3104, April 2020. ISSN 2374-3468. doi: 10.1609/aaai.v34i04.5705. Number: 04.

Ahmed, K., Teso, S., Chang, K.-W., Van den Broeck, G., and Vergari, A. Semantic Probabilistic Layers for Neuro-Symbolic Learning. In *Advances in Neural Information Processing Systems*, May 2022.

Ahmed, K., Chang, K.-W., and Van den Broeck, G. Semantic Strengthening of Neuro-Symbolic Learning. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pp. 10252–10261. PMLR, April 2023. ISSN: 2640-3498.

Avellaneda, F. A short description of the solver EvalMaxSAT. *MaxSAT Evaluation*, 8, 2020.

Badreddine, S., Garcez, A. d., Serafini, L., and Spranger, M. Logic Tensor Networks. *Artificial Intelligence*, 303: 103649, February 2022. ISSN 00043702. doi: 10.1016/j.artint.2021.103649.

Belle, V., Van den Broeck, G., and Passerini, A. Hashing-based approximate probabilistic inference in hybrid domains. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 141–150. AUAI PRESS, 2015.

Bengio, Y., Léonard, N., and Courville, A. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation, August 2013.

Bengio, Y., Lahlou, S., Deleu, T., Hu, E. J., Tiwari, M., and Bengio, E. GFlowNet Foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023. ISSN 1533-7928.

Chakraborty, S., Fremont, D., Meel, K., Seshia, S., and Vardi, M. Distribution-Aware Sampling and Weighted Model Counting for SAT. *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1), June 2014. ISSN 2374-3468. doi: 10.1609/aaai.v28i1.8990. Number: 1.

Chakraborty, S., Fried, D., Meel, K. S., and Vardi, M. Y. From Weighted to Unweighted Model Counting. In *IJCAI*, pp. 689–695, 2015.

Chakraborty, S., Meel, K. S., and Vardi, M. Y. Algorithmic improvements in approximate counting for probabilistic inference: from linear to logarithmic SAT calls. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pp. 3569–3576, New York, New York, USA, July 2016. AAAI Press. ISBN 978-1-57735-770-4.

Chakraborty, S., Meel, K. S., and Vardi, M. Y. Chapter 26. Approximate Model Counting. In *Handbook of Satisfiability*, pp. 1015–1045. IOS Press, 2021. doi: 10.3233/FAIA201010.

Chavira, M. and Darwiche, A. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6): 772–799, April 2008. ISSN 0004-3702. doi: 10.1016/j.artint.2007.11.002.

De Smet, L., Sansone, E., and Zuidberg Dos Martires, P. Differentiable Sampling of Categorical Distributions Using the CatLog-Derivative Trick. In *Advances in Neural Information Processing Systems*, November 2023.

Derkinderen, V., Manhaeve, R., Zuidberg Dos Martires, P., and De Raedt, L. Semirings for probabilistic and neuro-symbolic logic programming. *International Journal of Approximate Reasoning*, pp. 109130, January 2024. ISSN 0888-613X. doi: 10.1016/j.ijar.2024.109130.

Domshlak, C. and Hoffmann, J. Probabilistic Planning via Heuristic Forward Search and Weighted Model Counting. *Journal of Artificial Intelligence Research*, 30:565–620, December 2007. ISSN 1076-9757. doi: 10.1613/jair.2289.

Ermon, S., Gomes, C., and Selman, B. Uniform solution sampling using a constraint solver as an oracle. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, UAI'12, pp. 255–264, Arlington, Virginia, USA, August 2012. AUAI Press. ISBN 978-0-9749039-8-9.

Fichte, J. K., Hecher, M., and Hamiti, F. The Model Counting Competition 2020. *ACM Journal of Experimental Algorithmics*, 26:13:1–13:26, 2021. ISSN 1084-6654. doi: 10.1145/3459080.

Friedman, T. and Van den Broeck, G. Approximate Knowledge Compilation by Online Collapsed Importance Sampling. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Garcez, A., Gori, M., Lamb, L., Serafini, L., Spranger, M., and Tran, S. Neural-Symbolic Computing: An Effective Methodology for Principled Integration of Machine Learning and Reasoning. *FLAP*, May 2019.

Giunchiglia, E., Stoian, M. C., Khan, S., Cuzzolin, F., and Lukasiewicz, T. ROAD-R: The Autonomous Driving Dataset with Logical Requirements. *Machine Learning*, May 2023. ISSN 0885-6125, 1573-0565. doi: 10.1007/s10994-023-06322-z.

Gogate, V. and Dechter, R. SampleSearch: Importance sampling in presence of determinism. *Artificial Intelligence*, 175(2):694–729, February 2011. ISSN 0004-3702. doi: 10.1016/j.artint.2010.10.009.

Golia, P., Soos, M., Chakraborty, S., and Meel, K. S. Designing Samplers is Easy: The Boon of Testers. In *2021 Formal Methods in Computer Aided Design (FMCAD)*, pp. 222–230, October 2021. doi: 10.34727/2021/isbn.978-3-85448-046-4_31. ISSN: 2708-7824.

Gribkoff, E., Van den Broeck, G., and Suciu, D. Understanding the complexity of lifted inference and asymmetric Weighted Model Counting. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, UAI'14, pp. 280–289, Arlington, Virginia, USA, July 2014. AUAI Press. ISBN 978-0-9749039-1-0.

Hitzler, P. *Neuro-Symbolic Artificial Intelligence: The State of the Art*. IOS Press, January 2022. ISBN 978-1-64368-245-7.

Holtzen, S., Van den Broeck, G., and Millstein, T. Scaling exact inference for discrete probabilistic programs. *Proceedings of the ACM on Programming Languages*, 4(OOPSLA):140:1–140:31, November 2020. doi: 10.1145/3428208.

Huang, J., Li, Z., Chen, B., Samel, K., Naik, M., Song, L., and Si, X. Scallop: From Probabilistic Deductive Databases to Scalable Differentiable Reasoning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 25134–25145. Curran Associates, Inc., 2021.

Jang, E., Gu, S., and Poole, B. Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, November 2016.

Jerrum, M. R., Valiant, L. G., and Vazirani, V. V. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43: 169–188, January 1986. ISSN 0304-3975. doi: 10.1016/0304-3975(86)90174-X.

Karp, R. M., Luby, M., and Madras, N. Monte-Carlo approximation algorithms for enumeration problems. *Journal of Algorithms*, 10(3):429–448, September 1989. ISSN 0196-6774. doi: 10.1016/0196-6774(89)90038-2.

Kimmig, A., Santos Costa, V., Rocha, R., Demoen, B., and De Raedt, L. On the Efficient Execution of ProbLog Programs. In Garcia de la Banda, M. and Pontelli,

E. (eds.), *Logic Programming*, Lecture Notes in Computer Science, pp. 175–189, Berlin, Heidelberg, 2008. Springer. ISBN 978-3-540-89982-2. doi: 10.1007/978-3-540-89982-2_22.

Koller, D. and Friedman, N. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, July 2009. ISBN 978-0-262-01319-2.

Kool, W., Hoof, H. v., and Welling, M. Buy 4 REINFORCE Samples, Get a Baseline for Free! April 2019.

Lagniez, J.-M. and Marquis, P. An improved decision-DNNF compiler. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pp. 667–673, Melbourne, Australia, August 2017. AAAI Press. ISBN 978-0-9992411-0-3.

Li, Z., Yao, Y., Chen, T., Xu, J., Cao, C., Ma, X., and Lü, J. Softened Symbol Grounding for Neuro-symbolic Systems. In *International Conference on Learning Representations*, September 2022.

Maddison, C. J., Mnih, A., and Teh, Y. W. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *International Conference on Learning Representations*, November 2016.

Manhaeve, R., Dumancic, S., Kimmig, A., Demeester, T., and De Raedt, L. DeepProbLog: Neural Probabilistic Logic Programming. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Manhaeve, R., Marra, G., and De Raedt, L. Approximate Inference for Neural Probabilistic Logic Programming. In *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 475–486, 2021.

Marra, G., Dumančić, S., Manhaeve, R., and De Raedt, L. From Statistical Relational to Neurosymbolic Artificial Intelligence: a Survey. *Artificial Intelligence*, pp. 104062, January 2024. ISSN 0004-3702. doi: 10.1016/j.artint.2023.104062.

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. Monte Carlo gradient estimation in machine learning. *The Journal of Machine Learning Research*, 21(1):132:5183–132:5244, January 2020. ISSN 1532-4435.

Niepert, M., Minervini, P., and Franceschi, L. Implicit MLE: Backpropagating Through Discrete Exponential Family Distributions. In *Advances in Neural Information Processing Systems*, volume 34, pp. 14567–14579. Curran Associates, Inc., 2021.

Renkens, J., Van den Broeck, G., and Nijssen, S. k-Optimal: a novel approximate inference algorithm for ProbLog. *Machine Learning*, 89(3):215–231, December 2012. ISSN 0885-6125, 1573-0565. doi: 10.1007/s10994-012-5304-9.

Renkens, J., Kimmig, A., Van den Broeck, G., and De Raedt, L. Explanation-Based Approximate Weighted Model Counting for Probabilistic Logics. *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1), June 2014. ISSN 2374-3468. doi: 10.1609/aaai.v28i1.9067. Number: 1.

Roth, D. On the hardness of approximate reasoning. *Artificial Intelligence*, 82(1):273–302, April 1996. ISSN 0004-3702. doi: 10.1016/0004-3702(94)00092-1.

Soos, M. and Meel, K. S. BIRD: Engineering an Efficient CNF-XOR SAT Solver and Its Applications to Approximate Model Counting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):1592–1599, July 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33011592. Number: 01.

Soos, M., Gocht, S., and Meel, K. S. Tinted, Detached, and Lazy CNF-XOR Solving and Its Applications to Counting and Sampling. In Lahiri, S. K. and Wang, C. (eds.), *Computer Aided Verification*, Lecture Notes in Computer Science, pp. 463–484, Cham, 2020. Springer International Publishing. ISBN 978-3-030-53288-8. doi: 10.1007/978-3-030-53288-8_22.

Stockmeyer, L. The complexity of approximate counting. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, STOC '83, pp. 118–126, New York, NY, USA, December 1983. Association for Computing Machinery. ISBN 978-0-89791-099-6. doi: 10.1145/800061.808740.

Suciu, D., Olteanu, D., Ré, C., and Koch, C. *Probabilistic Databases*. Synthesis Lectures on Data Management. Springer International Publishing, Cham, 2011. ISBN 978-3-031-00751-4 978-3-031-01879-4. doi: 10.1007/978-3-031-01879-4.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

Valiant, L. G. The Complexity of Enumeration and Reliability Problems. *SIAM Journal on Computing*, 8(3):410–421, August 1979. ISSN 0097-5397. doi: 10.1137/0208032. Publisher: Society for Industrial and Applied Mathematics.

van Krieken, E., Acar, E., and van Harmelen, F. Analyzing Differentiable Fuzzy Logic Operators. *Artificial Intelligence*, 302:103602, January 2022. ISSN 0004-3702. doi: 10.1016/j.artint.2021.103602.

van Krieken, E., Thanapalasingam, T., Tomczak, J. M., Harmelen, F. V., and Teije, A. T. A-NeSI: A Scalable Approximate Method for Probabilistic Neurosymbolic Inference. In *Advances in Neural Information Processing Systems*, November 2023.

van Krieken, E., Minervini, P., Ponti, E. M., and Vergari, A. On the Independence Assumption in Neurosymbolic Learning. In *International Conference on Machine Learning*, April 2024.

Verreet, V., De Smet, L., and Sansone, E. EXPLAIN, AGREE and LEARN: A Recipe for Scalable Neural-Symbolic Learning, 2023.

Wei, W. and Selman, B. A New Approach to Model Counting. In Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G., Bacchus, F., and Walsh, T. (eds.), *Theory and Applications of Satisfiability Testing*, volume 3569, pp. 324–339. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. ISBN 978-3-540-26276-3 978-3-540-31679-4. doi: 10.1007/11499107_24. Series Title: Lecture Notes in Computer Science.

Xu, J., Zhang, Z., Friedman, T., Liang, Y., and Van den Broeck, G. A Semantic Loss Function for Deep Learning with Symbolic Knowledge. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 5502–5511. PMLR, July 2018. ISSN: 2640-3498.

Yang, Z., Ishay, A., and Lee, J. NeurASP: embracing neural networks into answer set programming. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20, pp. 1755–1762, Yokohama, Yokohama, Japan, January 2021. ISBN 978-0-9992411-6-5.

Zuidberg Dos Martires, P. Neural Semirings. 2021.

A. Example of Baysian Networks as Weighted Model Counting

As a small example, consider the context of a Bayesian network with a conditional probability $P(B|A)$ and independent probabilistic fact $P(A)$, over probabilistic Boolean variables A and B. This can be encoded into a WMC formula ϕ':

$$b \iff \big(a \wedge \theta_{b|a}\big) \vee \big(\neg a \wedge \theta_{b|\neg a}\big), \tag{3}$$

with the weights of $P(B|A)$ attached to newly introduced variables $\theta_{b|a}$ and $\theta_{b|\neg a}$. As is common, we can set $w(b) = w(\neg b) = 1$. The probability of B, for instance, is then $P(B) = WMC(\phi' \wedge b)$.

We limit our study to WMC problems where the weights correspond to a Bernoulli distribution. The example above, where $w(b) = w(\neg b) = 1$, can be encoded into this setting by instead using $w(b) = w(\neg b) = 0.5$ and normalizing the WMC by a factor 2^u, with u the number of variables whose original weights were 1. We do remark that this affects the tractability of additive (ϵ, δ)-bounds.

Categorical variables can be modeled using exclusively Boolean variables. For example, without using $w(\cdot) = 1$, suppose A takes values a_1, a_2 or a_3:

$$\big(a_1 \iff \theta_{a_1}\big) \wedge \big(a_2 \iff \neg a_1 \wedge \theta_{a_2|\neg a_1}\big) \wedge \big(a_3 \iff \neg a_1 \wedge \neg a_2\big) \tag{4}$$

The weights of a_i are 0.5 (using the method above), and the other weights are as expected: $w(\theta_{a_1}) = P(a_1)$, $w(\neg\theta_{a_1}) = 1 - w(\theta_{a_1})$, $w(\theta_{a_2|\neg a_1}) = \frac{P(A=a_2)}{P(A\neq a_1)}$ and $w(\neg\theta_{a_2|\neg a_1}) = 1 - w(\theta_{a_2|\neg a_1})$.

B. Variance of IndeCateR

Consider the single-sample IndeCateR estimator (De Smet et al., 2023) for $\partial\text{WMC}(\phi)/\partial w(x)$.

$$C = \mathbb{1}(I_0 \wedge x \models \phi) - \mathbb{1}(I_1 \wedge \neg x \models \phi)$$

By considering Theorem 3.1, it follows that $\text{Var}[C] \leq \mathbb{E}[C^2] = (\text{WMC}(\phi \mid x) - \text{WMC}(\phi \mid \neg x))^2 \leq 1$. Moreover, this variance is maximized when the weights w are binary. If we also know that the weights are bounded by $[t, 1-t]$ and ϕ has a single model, we achieve the much stronger result that $\text{Var}[C] \leq (1-t)^{\text{var}(\phi)}$. Although IndeCateR has vanishingly small variance on formulas with a single model, IndeCateR will almost never sample a model here and simply return zero gradients. This motivates that variance is not an ideal lens for looking at the quality of a gradient estimator for WMC.

C. Proofs

Theorem 4.1

Proof. We need to show that the following bound can be computed in polynomial time:

$$P(|(y - \hat{y})/y| > \epsilon) \leq \delta \quad \text{where } y = \frac{\partial\text{WMC}(\phi, w)}{\partial w(x)} = \text{WMC}(\phi \mid x) - \text{WMC}(\phi \mid \neg x)$$

As the theorem assumes that there is an implicant π with $x \in \pi$, it follows that:

$$\begin{aligned} y &= \text{WMC}(\phi \mid x) - \text{WMC}(\phi \mid \neg x) \\ &\geq \text{WMC}(\pi \mid x) - \text{WMC}(\phi \mid \neg x) \\ &\geq \text{WMC}(\pi) - \text{WMC}(\phi \mid \neg x) \\ &\geq c(\epsilon, \delta) \end{aligned}$$

In the last step, we use the assumed inequality of the theorem. So in summary, the assumption of the theorem implies that we know that $y \geq c(\epsilon, \delta)$. We also know that $\hat{y} \in [-1, 1]$. It follows that we can estimate it in polynomial time with sampling. To get a concrete expression for $c(\epsilon, \delta)$ and derive a bound on the number samples, we can use e.g. Hoefdding bounds. □

Theorem 4.3

Proof. Suppose ϕ only has a single model. This means that $y = \partial\text{WMC}(\phi)/\partial w(x) = \text{WMC}(M \mid x) = p(M)/w(x)$, assuming w.l.o.g. that $x \in M$. To ensure tractability by interpretation sampling, we need $y > c(\epsilon, \delta)$ (Karp et al., 1989). Because ϕ is τ-supervised, we have $P(M) \leq 2^{\tau - \text{var}(\phi)}$. When we combine these facts, we get that $w(x) \cdot c(\epsilon, \delta) < 2^{\tau - \text{var}(\phi)}$, which is equivalent to $\log_2 w(x) c(\epsilon, \delta) + \text{var}(\phi) < \tau$. To simplify the theorem, we set $c'(\epsilon, \delta) = -\log_2 w(x) c(\epsilon, \delta)$, which brings use to $\text{var}(\phi) - c'(\epsilon, \delta) < \tau$. □

Theorem 5.2

Proof. We write X for the random variable of WeightME.

$$
\begin{aligned}
\mathbb{E}_M[X] &= \frac{1}{w(x)} \mathbb{E}_M[\mathbb{1}(x \in M)] - \frac{1}{w(\neg x)} \mathbb{E}_M[\mathbb{1}(x \notin M)] \\
&= \Big(\sum_M \frac{\mathbb{1}(x \in M) P(M)}{w(x) \text{WMC}(\phi)} \Big) - \Big(\sum_M \frac{\mathbb{1}(x \notin M) P(M)}{w(\neg x) \text{WMC}(\phi)} \Big) \\
&= \frac{\text{WMC}(\phi \wedge x)}{w(x) \text{WMC}(\phi)} - \frac{\text{WMC}(\phi \wedge \neg x)}{w(\neg x) \text{WMC}(\phi)} \\
&= \frac{\text{WMC}(\phi \mid x)}{\text{WMC}(\phi)} - \frac{\text{WMC}(\phi \mid \neg x)}{\text{WMC}(\phi)} \\
&= \frac{1}{\text{WMC}(\phi)} \cdot \frac{\partial \text{WMC}(\phi)}{\partial w(x)} = \frac{\partial \log \text{WMC}(\phi)}{\partial w(x)}
\end{aligned}
$$

□

Theorem 5.3

Proof. We introduce an random variable T for $\mathbb{1}(x \in M)$, where $\mathbb{E}_M[T] = \text{WMC}(\phi \wedge x)/\text{WMC}(\phi)$. Using T, single-sample WeightME can be written as

$$
\begin{aligned}
X &= \frac{1}{w(x)} T - \frac{1}{w(\neg x)} (1 - T) \\
&= T \left(\frac{1}{w(x)} + \frac{1}{w(\neg x)} \right) - \frac{1}{w(\neg x)} \\
&= \frac{T}{w(x) w(\neg x)} - \frac{1}{w(\neg x)}
\end{aligned}
$$

Remember that the theorem assumes $w(x) \neq 0$ and $w(\neg x) \neq 0$. Next, we look for an (ϵ, δ)-approximation for WeightME, i.e. we again need to prove that

$$
P(|(y - \hat{y})/y| > \epsilon) \leq \delta \quad \text{where } y = \frac{\partial \text{WMC}(\phi, w)}{\partial w(x)} = \text{WMC}(\phi \mid x) - \text{WMC}(\phi \mid \neg x)
$$

We can write WeightME with s weighted model samples as the estimator $X' = \sum_{i=1}^{s} \frac{X_i}{s}$ where $\mathbb{E}[X'] = \mathbb{E}[X] = y/\text{WMC}(\phi)$ and $\hat{y} = \text{WMC}(\phi) X'$.

$$
\begin{aligned}
P\left(\left| \frac{\hat{y} - y}{y} \right| > \epsilon \right) &= P\left(\left| \frac{X' - \mathbb{E}[X']}{\mathbb{E}[X']} \right| > \epsilon \right) \\
&= P\left(|T' - \mathbb{E}[T']| > \epsilon |\mathbb{E}[T'] - w(x)| \right) \\
&\leq 2 \exp(-2 s \epsilon^2 (\mathbb{E}[T'] - w(x))^2)
\end{aligned}
$$

In the last step, we use Hoeffding's inequality. This is possible as T_i is bounded by $\left[0, \frac{1}{s}\right]$. We can work out this expression

to get a concrete lower bound on the number of samples.

$$\begin{aligned}
\delta &= -\exp(-2s\epsilon^2(\mathbb{E}[T'] - w(x))^2) \\
\log\frac{2}{\delta} &= 2s\epsilon^2(\mathbb{E}[T'] - w(x))^2 \\
s &= \frac{1}{2\epsilon^2(\mathbb{E}[T'] - w(x))^2}\log\frac{2}{\delta} \\
s &> \frac{1}{2\epsilon^2\lambda^2}\log\frac{2}{\delta} = c(\epsilon,\delta)/\lambda^2
\end{aligned}$$

The last line uses the assumption of the theorem. □

Approximate weighted model sampling

When we use an approximate weighted model sampler for WeightME, we introduce some bias. However, we can still get bounds on the approximation, if the approximate sampler has guarantees. An (ϵ, δ)-approximate sampler will sample a model M of ϕ with a probability between $p(M)/\text{WMC}(\phi)(1-\epsilon)$ and $p(M)/\text{WMC}(\phi)(1+\epsilon)$, or fail to sample a model with probability less than δ. So in other words, WeightME with an (ϵ, δ)-approximate sampler is bounded between $(1-\epsilon)\,\mathbb{E}[X]$ and $(1+\epsilon)\,\mathbb{E}[X]$, and the bias on the gradient is at most $\epsilon \cdot \nabla \log \text{WMC}(\phi)$. Moreover, we retain the probabilistic guarantee of Theorem 5.3.

D. Extra Baselines

The CatLog-trick (De Smet et al., 2023) applies the same decomposition as Theorem 3.1 to create gradient estimators:

$$\frac{\partial \text{WMC}(\phi)}{\partial w(x)} = \text{WMC}(\phi \mid x) - \text{WMC}(\phi \mid \neg x)$$

We can in principle use any other approximate method to compute $\text{WMC}(\phi|x)$ and $\text{WMC}(\phi|\neg x)$. When these are unbiased, it can be seen as a Rao-Blackwellization. This method does introduce a linear increase in complexity in the number of variables. In Figure 4, we include the CatLog-trick in combination with the product t-norm and Gumbel-softmax estimator.

Semantic strengthening needs to compare all clause pairs in order to choose which pairs to conjoin exactly. So it has a quadratic complexity in the number of clauses. As this becomes problematic on large formulas, we also consider the combination of sampling with fuzzy t-norms. Here, we keep sampling as long as we have models. The remaining conjunctions are calculated with the t-norm.

E. Additional experiments

In Figure 5, we repeat the optimization experiment. However, we now provide 90% accurate concept supervision. This means that we pick a model for each formula, and set the weight correctly for 90% of the variables. Although the results are a clear improvement on Figure 4, still none of the methods manage to optimize across the board. This confirms the result in Theorem 4.3.

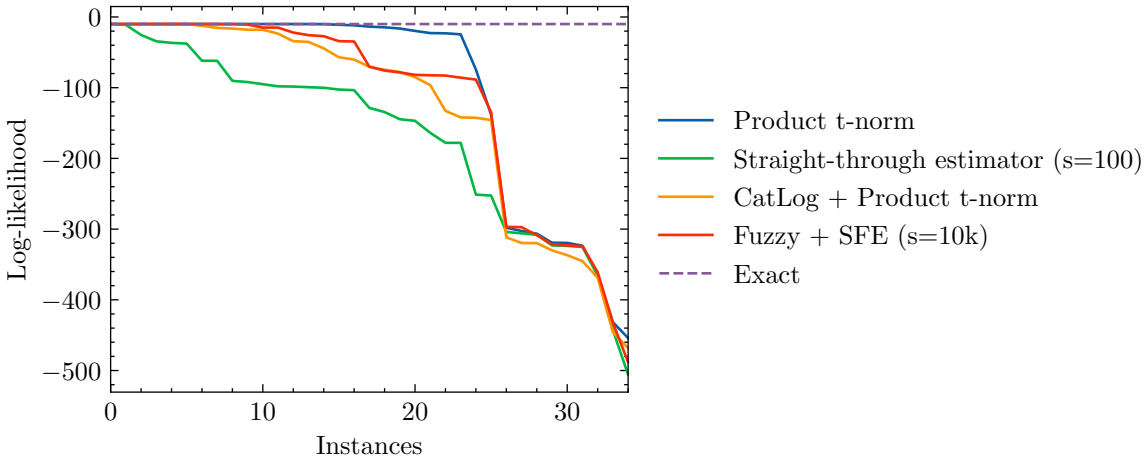


Figure 5: Maximum negative log-likelihood achieved by the various biased gradient approximations, sorted from best to worst. The benchmarks are 33 easy instances from the Model Counting Competition. Higher is better. All weights are initialized such that 90% of weights are already correct for a certain model.