Differentiation and Weighted Model Integration

Pedro Zuidberg Dos Martires

KU Leuven, Belgium
firstname.lastname@cs.kuleuven.be

Abstract. Stochastic gradient descent (SGD), especially in combination with auto-differentiation, has been the prime working horse of deep learning and has helped the field to rise to the most prominent spot of machine learning. Gradient based methods have also seen deployment in the field of probabilistic inference. However, optimization by differentiation in the field of probabilistic inference has until now only been targeted towards problems with either discrete or continuous random variables. In this note we show how to perform gradient based optimization on discrete-continuous probabilistic models, expressed as weighted model integration problems, by means of auto-differentiation. This provides a new powerful tool for inference, learning and optimization on (deep) discrete-continuous models.

Keywords: Probabilistic Inference · Satisfiability Modulo Theories · Auto-Differentiation.

1 Introduction

Weighted model integration (WMI) [1] is the task of performing probabilistic inference over a model defined as a logical structure. This means that, as such, WMI problems are not differentiable in the common sense. However, in [19] it was shown that WMI problems can be compiled into arithmetic circuits (AC) [5], which do constitute a differentiable structure. Deep ACs have recently been used to perform tasks nowadays more commonly situated in the neural domain [14]. Closely related to ACs are sum-product networks [16], which have also been shown to be able to perform tasks associated with neural networks [3].

The main concern of probabilistic inference is to efficiently perform integrals and summations over satisfying assignment to the random variables of a probabilistic model, however differentiation has also been a valuable tool for probabilistic inference, especially when dealing with continuous random variables. Differentiation is for example used when performing variational inference [10,17,18] or when performing Hamilton Monte Carlo inference [6,9]. Although not as important for the discrete domain as for continuous domain, differentiation has also sparsely been deployed to probabilistic inference with discrete random variables [4,8].

Even though differentiation has received its fair share of attention in the field of probabilistic inference for discrete and continuous random variables separately, this is not true for the hybrid domain. In this note we start to repair this omission and show how to perform differentiation over arithmetic circuits compiled from WMI problems by using off-the-shelf auto-differentiation software [15].

2 Weighted Model Integration in a Nutshell

In this section we are going to formally introduce weighted model integration. First we define the logical language that allows us to write down a WMI problem.

Definition 1. *(SMT($\mathcal{RA}$) (satisfiability modulo theories over the real arithmetics)) Let $\mathbb{R}$ denote the set of reals, $\mathbb{B} = \{\bot, \top\}$ the set of Boolean values, let B be a set of M Boolean and X a set of N real variables. An atomic formula is an expression of the form $g(X) \bowtie c$, where $c \in \mathbb{R}$, $\bowtie \in \{=, \neq, \geq, \leq, >, <\}$, and $g : \mathbb{R}^N \rightarrow \mathbb{R}$. We then define SMT($\mathcal{RA}$) theories as Boolean combinations (by means of the standard Boolean operators $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$) of Boolean variables $b \in B$ and of atomic formulas over X.*

We can now define the weighted model integral performed over a SMT($\mathcal{RA}$) formula.

Definition 2. *(WMI) Given a set* $\mathbf{b}$ *of M Boolean variables,* $\mathbf{x}$ *of N real variables, a weight function $w : \mathbb{B}^M \times \mathbb{R}^N \rightarrow \mathbb{R}_{\geq 0}$, and a support ϕ, in the form of an SMT formula, over* $\mathbf{b} \cup \mathbf{x}$*, the weighted model integral is:* $\mathrm{WMI}(\phi, w|\mathbf{x}, \mathbf{b}) = \sum_{\mathbf{b}_\mathcal{I} \in \mathcal{I}_\mathbf{b}(\phi)} \int_{\mathcal{I}_\mathbf{x}(\phi^{\mathbf{b}_\mathcal{I}})} w(\mathbf{x}, \mathbf{b}_\mathcal{I}) d\mathbf{x}$.

$\mathcal{I}_\mathbf{b}(\phi)$ and $\mathcal{I}_\mathbf{x}(\phi^{\mathbf{b}_\mathcal{I}}))$ denote the set of satisfying interpretations (or models) of ϕ and $\phi^{\mathbf{b}_\mathcal{I}}$ in function of the sets of variables $\mathbf{b}$ and $\mathbf{x}$ respectively (see [19, Defintion 2]). In [19] the authors also show that the weighted model integral can be rewritten as an integral of a sum-product multiplied by a probability density function:

$$\mathrm{WMI}(\phi, w|\mathbf{x}, \mathbf{b}) = \int \sum_{\mathbf{b}_\mathcal{I} \in \mathcal{I}_{\mathbf{b},\mathbf{b}_a}(\phi_a)} \prod_{b_i \in \mathbf{b}_\mathcal{I}} \alpha_{b_i}(\mathbf{x}) w_x(\mathbf{x}) d\mathbf{x} \tag{1}$$

In Equation 1 ϕ_a denotes a logical formula where all the atomic formulas (e.g. $(20<x)$) in ϕ have been replaced by fresh Boolean variables - ϕ_a is propositional. The set of these fresh Boolean variables is denoted by $\mathbf{b}_a$. α_{b_i} is the so-called labeling function and maps a Boolean variable b_i to a real number between in $[0, 1]$ in function of $\mathbf{x}$. A Boolean in the set $\mathbf{b}$ is mapped for example to 0.2, meaning that this Boolean is `True` with probability 0.2. A Boolean in the set $\mathbf{b}_x$ is mapped to to the Iverson bracket corresponding to its originating atomic SMT formula. For instance: $\alpha_{(20<x)}(x) = [20<x]$, which is 1 whenever the condition in the Iverson bracket is satisfied and 0 otherwise.

We will denote the sum-product in the weighted model integral from here on by Ψ. Inspecting Equation 1, we observe that the weighted model integral is in fact the expected value of Ψ with regards to the probability density function w_x as already pointed out in [19, Theorem 3]: $WMI(\phi, w|X, B) = \mathbb{E}_{w_x(\mathbf{x})}[\Psi(\mathbf{x})]$.

Example 1. Consider the SMT theory $\texttt{broken} \leftrightarrow (\texttt{no\_cool} \wedge (\texttt{t} > 20)) \vee (\texttt{t} > 30)$, where `broken` and `no_cool` are a Boolean variables and `t` is a real-valued variable. SMT then answers the question whether or not `broken` is satisfiable, i.e. whether or not there is a satisfying assignment to the formula for the variables `no_cool` and `t`. In the case that a probability distribution is given over the variables, we can now ask the question of the probability of the formula being satisfied. Let's take $p(\texttt{no\_cool}) = 0.01$ and $\texttt{t} \sim \mathcal{N}_t(20, 5)$. WMI then answers the question of how probable it is for the SMT formula to be satisfied.

$$\begin{aligned} p(\texttt{broken}) &= \int (0.01[\texttt{t}>20][\texttt{t}\leq 30] + [\texttt{t}>30])\, \mathcal{N}_\texttt{t}(20, 5) d\texttt{t} \\ &= 0.01 \int_{20<\texttt{t}\leq 30} \mathcal{N}_\texttt{t}(20, 5) d\texttt{t} + \int_{\texttt{t}>30} \mathcal{N}_\texttt{t}(20, 5) d\texttt{t} \end{aligned}$$

Note that we had to take the conjugate of `t` > 30 in the first term of the first line to avoid double counting. For a more in depth presentation of WMI, you can consult [19].

3 Taking the Gradient

Differentiation in probabilistic inference frequently occurs in the form of taking the gradient with regards to some parameters of an expectation of a random variable. A prominent example would be variational inference through minimization of the Kullback-Leibler (KL) divergence by the means of SGD. Due to space constraints we will focus, however, on the smaller sister of the KL-divergence, namely the cross-entropy. The cross-entropy of the distributions p and q takes the following mathematical form: $\mathbb{E}_p[-\log q]$. p can for example be the true distribution of a random variable, which we do not know but that we can observe (and for which we have data points) and q is a model that we would like to match as closely as possible to the true distribution p. Assuming that q depends on the set of parameters $\boldsymbol{\theta}$ then minimizing the cross-entropy by taking the gradient with regards to $\boldsymbol{\theta}$ and incrementally performing the update step $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \nabla_{\boldsymbol{\theta}} \mathbb{E}_p[-\log q(\boldsymbol{\theta})]$ minimizes the cross entropy.

Proposition 1. *(Gradient of the cross-entropy)*

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_p[-\log q(\boldsymbol{\theta})] = \mathbb{E}_p[-\tfrac{1}{q(\boldsymbol{\theta})} \nabla_{\boldsymbol{\theta}} \textstyle\sum_{\mathbf{b}_{\mathcal{I}} \in \mathcal{I}_{\mathbf{b},\mathbf{b}_a}(\phi_a)} \prod_{b_i \in \mathbf{b}_{\mathcal{I}}} \alpha_{b_i}(\boldsymbol{\theta})] \tag{2}$$

$$= \mathbb{E}_p[-\tfrac{1}{q(\boldsymbol{\theta})} \textstyle\sum_{\mathbf{b}_{\mathcal{I}} \in \mathcal{I}_{\mathbf{b},\mathbf{b}_a}(\phi_a)} \sum_{b_i \in \mathbf{b}_{\mathcal{I}}} \nabla_{\boldsymbol{\theta}}(\alpha_{b_i}(\boldsymbol{\theta})) \prod_{b_j \in \mathbf{b}_{\mathcal{I}} \setminus \{b_i\}} \alpha_{b_j}(\boldsymbol{\theta})] \tag{3}$$

Where in Equation 2 we switched around the order of the expectation and the gradient, and replaced $q(\boldsymbol{\theta})$ by the sum-product from the weighted model integral. In Equation 3 we further pushed inside the gradient by applying the product rule. The labeling function α is now a labeling function over the parameters to optimize.

In [13] and [12] the authors come to the conclusion (through different means) that when taking the derivative over an indicator function one ends up with an expectation over a boundary term, which is equivalent to performing an integral over a sub-manifold[1] — a non-trivial task in the general case. We propose, instead of performing the hard surface integral in a sub-manifold, to use a convex relaxation of the indicator function. This is for example also done when one deals with 0/1-loss functions, which are known to be NP-hard to optimize [2,7]. Once, the indicator function (i.e. the label of an atomic SMT formula) is relaxed, for example through an exponential, the gradient is computed in a straightforward fashion through auto-differentiation using an off-the-shelf software package [15] in combination with the reparametrization trick [11] and gradient descent.

4 Conclusion

In this note we have outlined how discrete-continuous probabilistic inference in the form of WMI fits into a larger a larger body of literature of deep probabilistic models and we have shown how SGD, the working horse of deep learning, could be applied to probabilistic inference for WMI problems.

[1] In the one dimensional case the derivative of the indicator function is the Dirac delta function, with its root determining the sub-manifold.

References

1. Belle, V., Passerini, A., Van den Broeck, G.: Probabilistic Inference in Hybrid Domains by Weighted Model Integration. In: IJCAI. pp. 2770–2776 (2015)
2. Ben-David, S., Eiron, N., Long, P.M.: On the difficulty of approximately maximizing agreements. Journal of Computer and System Sciences **66**(3), 496–514 (2003)
3. Butz, C.J., Oliveira, J.S., dos Santos, A.E., Teixeira, A.L.: Deep convolutional sum-product networks. In: Proceedings of the 30th Conference on Artificial Intelligence. AAAI Press (2019)
4. Darwiche, A.: A differential approach to inference in bayesian networks. Journal of the ACM (JACM) **50**(3), 280–305 (2003)
5. Darwiche, A.: Modeling and reasoning with Bayesian networks. Cambridge University Press (2009)
6. Duane, S., Kennedy, A.D., Pendleton, B.J., Roweth, D.: Hybrid monte carlo. Physics letters B **195**(2), 216–222 (1987)
7. Feldman, V., Guruswami, V., Raghavendra, P., Wu, Y.: Agnostic learning of monomials by halfspaces is hard. SIAM Journal on Computing **41**(6), 1558–1590 (2012)
8. Gutmann, B., Kimmig, A., Kersting, K., De Raedt, L.: Parameter learning in probabilistic databases: A least squares approach. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 473–488. Springer (2008)
9. Hoffman, M.D., Gelman, A.: The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. Journal of Machine Learning Research **15**(1), 1593–1623 (2014)
10. Jordan, M.I., Ghahramani, Z., Jaakkola, T.S., Saul, L.K.: An introduction to variational methods for graphical models. Machine learning **37**(2), 183–233 (1999)
11. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: Proceedings of the 5th International Conference on Representation Learning (2014)
12. Lange, R.J.: Potential theory, path integrals and the laplacian of the indicator. Journal of High Energy Physics **2012**(11), 32 (2012)
13. Lee, W., Yu, H., Yang, H.: Reparameterization gradient for non-differentiable models. In: Advances in Neural Information Processing Systems. pp. 5553–5563 (2018)
14. Liang, Y., Van den Broeck, G.: Learning logistic circuits. A¬ A **1**, 3 (2019)
15. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. NIPS 2017 Workshop Autodiff (2017)
16. Poon, H., Domingos, P.: Sum-product networks: A new deep architecture. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). pp. 689–690. IEEE (2011)
17. Ranganath, R., Gerrish, S., Blei, D.: Black box variational inference. In: Artificial Intelligence and Statistics. pp. 814–822 (2014)
18. Wainwright, M.J., Jordan, M.I., et al.: Graphical models, exponential families, and variational inference. Foundations and Trends® in Machine Learning **1**(1–2), 1–305 (2008)
19. Zuidberg Dos Martires, P., Dries, A., De Raedt, L.: Exact and approximate weighted model integration withprobability density functions using knowledge compilation. In: Proceedings of the 30th Conference on Artificial Intelligence. AAAI Press (2019)