Unconstrained Monotonic Neural Networks

Antoine Wehenkel[1] and Gilles Louppe[1]

1: Department of Electrical Engineering and Computer Science, University of Liège

Abstract. Monotonic neural networks have recently been proposed as a way to define invertible transformations. These transformations can be combined into powerful autoregressive flows that have been shown to be universal approximators of continuous probability distributions. Architectures that ensure monotonicity typically enforce constraints on weights and activation functions, which enables invertibility but leads to a cap on the expressiveness of the resulting transformations. In this work, we propose the Unconstrained Monotonic Neural Network (UMNN) architecture based on the insight that a function is monotonic as long as its derivative is strictly positive. In particular, this latter condition can be enforced with a free-form neural network whose only constraint is the positiveness of its output. We evaluate our new invertible building block within a new autoregressive flow (UMNN-MAF) and demonstrate its effectiveness on density estimation experiments. We also illustrate the ability of UMNNs to improve variational inference.

Keywords: Density Estimator · Normalizing Flow · Invertible Networks · Unsupervised Learning

Motivation and contributions Monotonic neural networks have been known as powerful tools to build monotone models of a response variable with respect to individual explanatory variables [1, 2, 5]. Recently, strictly monotonic neural networks have also been proposed as a way to define invertible transformations. These transformations can be combined into effective autoregressive flows that can be shown to be universal approximators of continuous probability distributions. Architectures that ensure monotonicity, such as Neural Autoregressive Flows [4] and Block Neural Autoregressive Flows [3], typically enforce constraints on weight and activation functions, which enables invertibility but leads to a cap on the expressiveness of the resulting transformations. This does not necessarily impede universal approximation but typically requires either complex conditioners or a composition of multiple flows. We summarize our contributions as follows: 1) We introduce the Unconstrained Monotonic Neural Network (UMNN) architecture, a new reversible scalar transformation defined via a free-form neural network. 2) We combine UMNN transformations into an autoregressive flow (UMNN-MAF) and we demonstrate competitive or state-of-the-art results on

Copyright ©2019 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR-WS.org/Vol-2491/abstract54.pdf

benchmarks for normalizing flows. 3) We empirically illustrate the scalability of our approach by applying UMNN on high dimensional density estimation problems.

Unconstrained monotonic neural networks We show how to parameterize a strictly monotonic scalar function $F(x; \boldsymbol{\psi}) : \mathbb{R} \to \mathbb{R}$ without imposing strong constraints on the expressiveness of the hypothesis class. In UMNNs, we achieve this by only imposing the derivative $f(x; \boldsymbol{\psi}) = \frac{\partial F(x;\boldsymbol{\psi})}{\partial x}$ to remain of constant sign or, without loss of generality, to be strictly positive. As a result, we can parameterize the bijective mapping $F(x; \boldsymbol{\psi})$ via its strictly positive derivative $f(\cdot; \boldsymbol{\psi}) : \mathbb{R} \to \mathbb{R}_+$ as $F(x; \boldsymbol{\psi}) = \int_0^x f(t; \boldsymbol{\psi})\ \mathrm{d}t + F(0; \boldsymbol{\psi})$, where $F(0; \boldsymbol{\psi}) = \beta \in \mathbb{R}$ is a scalar. We make f arbitrarily complex using an unconstrained neural network whose output is forced to be strictly positive. $\boldsymbol{\psi}$ denotes the parameters of this neural network. In the full version of the paper, we show how the forward and backward passes for the integral can be efficiently implemented.

UMNN autoregressive transformations (UMNN-MAF) Normalizing flows are often expressed as a composition of autoregressive transformations $\boldsymbol{g}$, i.e., such that $\boldsymbol{g}(\boldsymbol{x}; \boldsymbol{\theta})$ can be rewritten as a vector of d scalar functions, $\boldsymbol{g}(\boldsymbol{x}; \boldsymbol{\theta}) = \left[g^1(x_1; \boldsymbol{\theta}) \dots g^i(\boldsymbol{x}_{1:i}; \boldsymbol{\theta}) \dots g^d(\boldsymbol{x}_{1:d}; \boldsymbol{\theta})\right]$, where $\boldsymbol{x}_{1:i} = \left[x_1 \dots x_i\right]^T$ is the vector including the i first elements of the full vector $\boldsymbol{x}$. The Jacobian of this function is lower triangular, which makes the computation of its determinant $\mathcal{O}(d)$. Enforcing the bijectivity of each component g^i is then sufficient to make $\boldsymbol{g}$ bijective as well. In this work, we combine UMNNs with an embedding of the conditioning variables to build invertible autoregressive functions g^i. Specifically, we define

$$g^i(\boldsymbol{x}_{1:i}; \boldsymbol{\theta}) = \int_0^{x_i} f^i(t, \boldsymbol{h}^i(\boldsymbol{x}_{1:i-1}; \boldsymbol{\phi}^i); \boldsymbol{\psi}^i)\ \mathrm{d}t + \beta^i(\boldsymbol{h}^i(\boldsymbol{x}_{1:i-1}; \boldsymbol{\phi}^i)),$$

where $\boldsymbol{h}^i(\cdot; \boldsymbol{\phi}^i) : \mathbb{R}^{i-1} \to \mathbb{R}^q$ is a q-dimensional neural embedding of the conditioning variables $\boldsymbol{x}_{1:i-1}$ and $\beta(\cdot)^i : \mathbb{R}^{i-1} \to \mathbb{R}$. Figure 1 summarizes the complete architecture and Figure 2 presents density estimation performed on multi-modal and discontinuous 2D distributions.

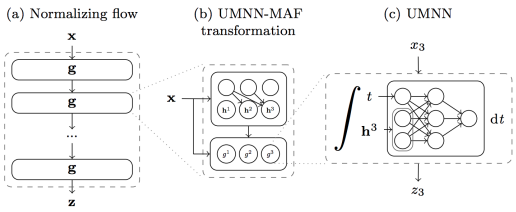

Fig. 1: **(a)** A normalizing flow made of repeated UMNN-MAF transformations $\boldsymbol{g}$ with identical architectures. **(b)** A UMNN-MAF which transforms a vector $\boldsymbol{x} \in \mathbb{R}^3$. **(c)** The UMNN network used to map x_3 to z_3 conditioned on the embedding $\boldsymbol{h}^3(\boldsymbol{x}_{1:2})$.

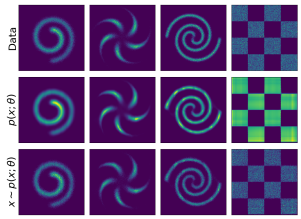

Fig. 2: **Top**: Samples from the empirical distribution $p(\boldsymbol{x})$. **Middle**: Learned density $p(\boldsymbol{x};\boldsymbol{\theta})$. **Bottom**: Samples by numerical inversion.

References

1. Archer, N.P., Wang, S.: Application of the back propagation neural network algorithm with monotonicity constraints for two-group classification problems. Decision Sciences **24**(1), 60–75 (1993)
2. Daniels, H., Velikova, M.: Monotone and partially monotone neural networks. IEEE Transactions on Neural Networks **21**(6), 906–917 (2010)
3. De Cao, N., Titov, I., Aziz, W.: Block neural autoregressive flow. arXiv preprint arXiv:1904.04676 (2019)
4. Huang, C.W., Krueger, D., Lacoste, A., Courville, A.: Neural autoregressive flows. In: International Conference on Machine Learning. pp. 2083–2092 (2018)
5. Sill, J.: Monotonic networks. In: Advances in neural information processing systems. pp. 661–667 (1998)