Probabilistic Flow Circuits: Towards Unified Deep Models for Tractable Probabilistic Inference Supplementary Material

Sahil Sidheekh[1] **Kristian Kersting**[2,3] **Sriraam Natarajan**[1]

[1]Erik Jonsson School of Engineering & Computer Science, The University of Texas at Dallas
[2]Department of Computer Science , TU Darmstadt
[3] Centre for Cognitive Science, TU Darmstadt, and Hessian Center for AI

In this supplementary material, we furnish further details pertaining to the theory and implementation that was left out in the main paper due to space constraints.

1 THEORETICAL RESULTS

Lemma 2. *τ-decomposability is a necessary condition for an SPTN to be decomposable.*

Proof. Let $\mathcal{C}_{\mathcal{SPTN}}$ be a decomposable sum-product transform the network. $\implies$ $\forall \mathcal{P} \in \mathcal{C}_{\mathcal{SPTN}}, \mathcal{P}$ is decomposable $\implies$ $\forall \mathcal{N}_i, \mathcal{N}_j \in ch(\mathcal{P}), i \neq j, \psi_{\mathcal{N}_i} \cap \psi_{\mathcal{N}_j} = \emptyset$. Now, let $\mathcal{T} \in \mathcal{C}_{\mathcal{SPTN}}$ be a transform node (and g be its associated transformation) that is not τ-decomposable. i.e. when defined over a product node $\mathcal{P}$, there exists at least one pair $\psi_{i'}, \psi_{j'} \in \{\psi_{\mathcal{N}_i}\}_{\mathcal{N}_i \in ch(\mathcal{P})}, i^{'} \neq j^{'}$, such that for $\boldsymbol{x} \in \mathbb{R}^{|\psi_P|}$ and $\boldsymbol{y} = g(\boldsymbol{x}), \boldsymbol{y}_{\psi_{i'}} \not\perp \boldsymbol{x}_{\psi_{j'}} \implies \Pi_{\psi_{i'}}(\boldsymbol{y}) = f(\boldsymbol{x}_{\psi_{j'}})$ for some function f. Thus, we have,

$$\begin{aligned}\mathcal{T}(\mathcal{P}(\boldsymbol{x})) &= \mathcal{P}(g(\boldsymbol{x}))|\det J_g| \\ &= \prod_{N_i \in ch(\mathcal{P})} \mathcal{N}_i(\Pi_{\psi_{\mathcal{N}_i}}(g(\boldsymbol{x})))|\det J_g|\end{aligned}$$

Thus the child $\mathcal{N}_{i'}$ of $\mathcal{P}$ computes a function over $\boldsymbol{x}_{\psi_{j'}} \implies \psi_{N_{i'}} \supset \psi_{N_{j'}} \implies \psi_{N_{i'}} \cap \psi_{N_{j'}} \neq \emptyset \implies \mathcal{P}$ is not decomposable, thus resulting in a contradiction. Thus, for a sum product transform network $\mathcal{C}_{\mathcal{SPTN}}$ to be decomposable, all transform nodes must be τ-decomposable.

□

Lemma 3. *A PFC ($\mathcal{C}_{\mathcal{F}}$) with leaf distributions defined using g_{lrs} transformations and a Student's-t distribution with $\nu = 3$ as the base distribution is a tractable model for (a) evidential inference if $\mathcal{C}_{\mathcal{F}}$ is smooth, (b) Marginal and conditional inference if $\mathcal{C}_{\mathcal{F}}$ is smooth and decomposable, (c) MAP inference if $\mathcal{C}_{\mathcal{F}}$ is smooth, decomposable, and deterministic.*

Proof. The tractability of evidential, marginal and conditional inference for $\mathcal{C}_{\mathcal{F}}$ follows trivially from the fact that $\mathcal{C}_{\mathcal{F}}$ is a probabilistic circuit and hence inherits the tractability offered by the circuit properties of a PC under the structural constraints of smoothness an decomposability. We elaborate this further below.

(a) **Evidential inference:** In order to tractably perform evidential inference, $\mathcal{C}_{\mathcal{F}}$ requires that the leaf nodes compute a valid probability density over its scope. A normalizing flow supports exact density evaluation using the change of variables formula and hence enables tractable evidential inference. Smoothness of $\mathcal{C}_{\mathcal{F}}$ further ensures that its sum nodes compute valid mixture densities. However, note that smoothness is not a necessary condition, as a non smooth PC can, in polynomial time, be converted to a smooth PC (Choi et al. [2020]).

(b) **Marginal and Conditional inference:** For a smooth and decomposable $\mathcal{C}_{\mathcal{F}}$, marginalizing out a variable X_i from its modeled density reduces to marginalizing out the corresponding leaf distribution. This is because marginalization of X_i involves integrating the model density over $\boldsymbol{val}(X_i)$, and as proved in Choi et al. [2020], the integral over the circuit reduces to integrals over the leaf distributions having X_i in their scope, when the circuit is smooth and decomposable. Note that each leaf nodes in $\mathcal{C}_{\mathcal{F}}$ represents a probability distribution over a single variable and marginalizing it out is equivalent to setting the corresponding leaf density to 1. Thus, $\mathcal{C}_{\mathcal{F}}$ supports tractable marginal inference. Also, the tractability of conditional inference naturally follows from the tractability of evidential and marginal inference.

(b) **MAP inference:** Along the same lines, computation of MAP queries for $\mathcal{C}_{\mathcal{F}}$ reduces to computing argmax over leaf densities if $\mathcal{C}_{\mathcal{F}}$ is smooth, decomposable and deterministic Choi et al. [2020]. Thus, if we can compute the mode of the distribution modeled by the leaf nodes, we can ensure tractability for MAP inference. For $x \in [x^i, x^{i+1}]$, let $\phi = \frac{(x - x^{i+1})}{x^{i+1} - x^i}$, and let g denote the linear rational spline transformation associated with the bin, which has the form

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

$g(\phi) = \frac{q(\phi)}{r(\phi)} = \frac{a_1\phi+b_1}{a_2\phi+b_2}$. Let S_t denote a Student's-t distribution with 3 degrees of freedom. The pdf of a Student's-t distribution with ν degrees of freedom is given by:

$$p(x;\nu) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1+\frac{x^2}{\nu}\right)^{-(\frac{\nu+1}{2})}$$

Thus, we have the following form for the density modeled at the leaf distributions,

$$\begin{aligned}
p(x) &= S_t(g(\phi)).|\frac{\partial g(\phi)}{\partial x}| \\
&= \frac{1}{(x^{i+1}-x^i)} S_t(g(\phi)).|\frac{\partial g(\phi)}{\partial \phi}| \\
&= C_1 S_t\left(\frac{q(\phi)}{r(\phi)}\right)(r(\phi))^{-2} \\
&= \frac{C_1\Gamma(2)}{\sqrt{3\pi}\Gamma(\frac{3}{2})}\left[1+\frac{1}{3}\left(\frac{q(\phi)}{r(\phi)}\right)^2\right]^{-2}(r(\phi))^{-2} \\
&= C_2\left[3r(\phi)+q(\phi)g(\phi)\right]^{-2}
\end{aligned}$$

Where, C_1, C_2 are constants. Thus, we have $\log p(x) = \log C_2 - 2\log(3r(\phi)+q(\phi)g(\phi))$. Now, $\log p(x)$ is maximized when $f(\phi) = 3r(\phi)+q(\phi)g(\phi)$ is minimized. Differentiating and equating to zero, we have,

$$\begin{aligned}
&3r'(\phi)+q'(\phi)g(\phi)+q(\phi)g'(\phi) = 0 \\
&\implies 3a_2r(\phi)^2 + a_1q(\phi)r(\phi) + (b_2a_1 - a_2b_1)q(\phi) = 0
\end{aligned}$$

Note that as $q(\phi), r(\phi)$ are linear in ϕ, the above equation is quadratic in ϕ. Thus, we can check if any of its real roots lie within the interval $[0,1]$. If it does, then the maximum density within that bin is given by the density at the root. If not, then the maximum occurs at either of the interval boundaries. Thus, we can compute the maximum within each bin analytically. The maximum density across all the bins gives the mode of the distribution.

Note that the above analysis also extends to compositions of LRS transformations as a composition of linear rational functions is a linear rational function .i.e for $g_1(\phi) = \frac{a_1\phi+b_1}{a_2\phi+b_2}$ and $g_2(\phi) = \frac{c_1\phi+d_1}{c_2\phi+d_2}$, we have,

$$\begin{aligned}
g_2(g_1(\phi)) &= \frac{c_1\left(\frac{a_1\phi+b_1}{a_2\phi+b_2}\right)+d_1}{c_2\left(\frac{a_1\phi+b_1}{a_2\phi+b_2}\right)+d_2} \\
&= \frac{(c_1a_1+d_1a_2)\phi + c_1b_1 + d_1b_2}{(c_2a_1+d_2a_2)\phi + c_2b_1 + d_2b_2}
\end{aligned}$$

is also a linear rational function. □

2 IMPLEMENTATION DETAILS

We implemented our code using pytorch, adapting from Peharz et al. [2020]. We used the Pyro probabilistic programming language Bingham et al. [2019] package that is built on top of pytorch to implement the linear rational spline transformations. The hyper parameters defining the structure of the PC in einsum networks [Peharz et al., 2020] are - the depth (D) of the circuit, the number of vector components (K) (i.e. the no. of leaf distributions per variable and the no. sum nodes in an einsum-layer) and the number of replica (R). We use the same PC architecture for both the EinsumNet and EinsumNet+LRS. There are two hyper parameters associated with the linear rational spline transformation - the no. of of intervals (I), the bounds (B) within which it is defined. Outside of $[-B, B]$ the transformation is defined to be identity. For a dataset with n features, we set depth $D = \max(1, \lfloor\log_2(n)\rfloor)$. We use end-to-end backpropagation and train all our models using an Adam optimizer, with a learning rate of $1e-3$. Further dataset specific details are summarized below:

3D Manifold Data We sampled $20,000$ data points for each of the 6 3D datasets, $10,000$ of which we used for training and $5,000$ each for validation and testing. We used $K = 10, R = 10$ as the underlying PC structure for each model on the 3D datasets. We used a single linear rational spline transformation with $I = 16, B = 20$ at the leaves of EinsumNet+LRS. We used a batch size of 100 and trained all models for 200 epochs. The learning curves of the models on the 3D datasets left out in the main paper is given in Figure 1.

UCI Tabular Datasets We used $K = 20, R = 20$ as the underlying PC structure for all models on the tabular datasets. We used a single linear rational spline transformation with $I = 16, B = 16$ at the leaves of EinsumNet+LRS. We used a batch size of 200 for the GAS, MINIBOONE and HEPMASS datasets, and a batch size of 500 for the POWER dataset. We trained all models for 100 epochs, early stopping if there is no improvement in the validation performance for over 5 epochs. The details regarding the no. of features, and no. of datapoints within in each split of the 4 UCI tabular datasets considered can be found in Papamakarios et al. [2017]. We also followed the same preprocessing for the datasets as given by Papamakarios et al. [2017].

Image Datasets We used the PD structure [Poon and Domingos, 2011, Peharz et al., 2020] with $\triangle = [7, 28]$ and $K = 20, R = 20$ to define the underlying PC structure for all the models on the two image datasets - MNIST and Fashion-MNIST. We used two linear rational spline transformations with $I = 16, B = 16$ at the leaves of EinsumNet+LRS. We used a batch size of 100 and trained all models for 50 epochs. As we consider continuous leaf distributions, we applied the logit transformations as done in Papamakarios et al. [2017] to make the data continuous.

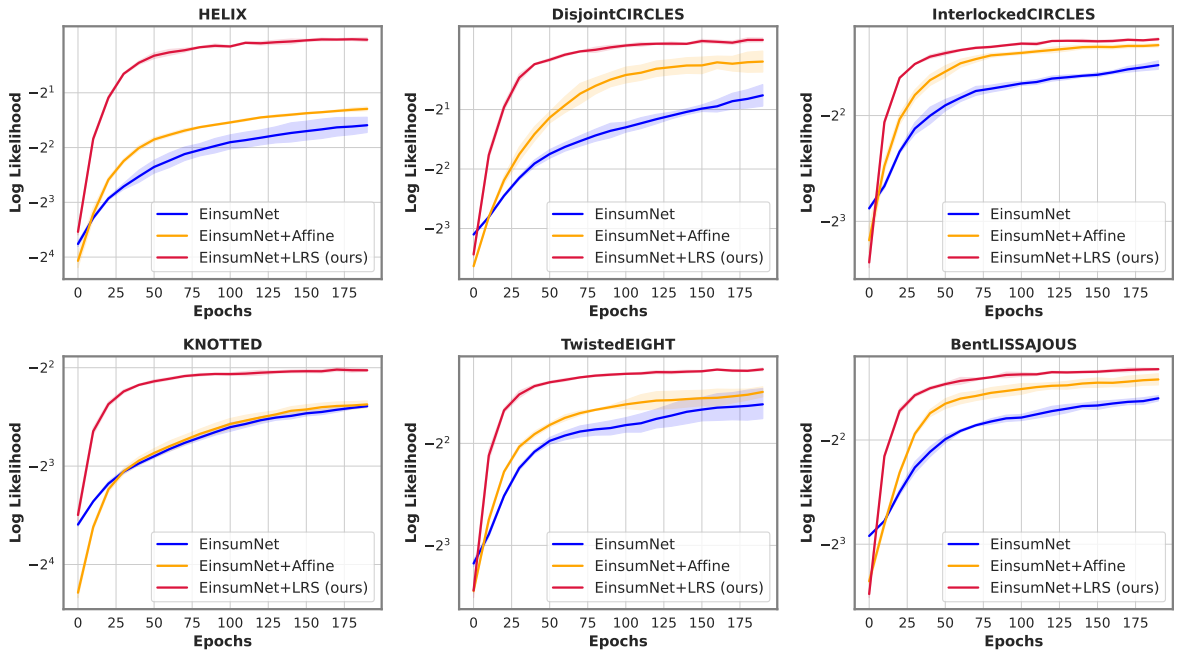

Figure 1: **Learning curves** of - (a) *Einsum Network* (b) *Einsum Network + Affine* transformations at the leaves and (c) *Einsum Network + LRS* transformations at the leaves on the **3D datasets**, in terms of average log-likelihood (***higher the better***) on the validation set across training epochs. The shaded regions depict the standard deviation across 3 independent trials. We can observe that *Einsum Network + LRS* achieves superior performance much faster than the other two models on all datasets.

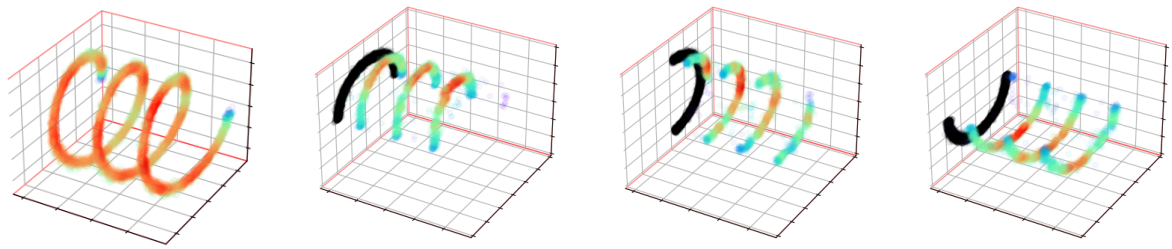

Figure 2: Controlled Sample Generation Using an EinsumNet+LRS trained on the Helix dataset. The tractability of EinsumNet+LRS allows generating data with certain properties, for e.g. the second, third and fourth subfigures show data generated such that its projection onto the XY plane is the black curve plotted.

References

Eli Bingham, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman. Pyro: Deep universal probabilistic programming. *The Journal of Machine Learning Research*, 20(1), 2019.

YooJung Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. 2020.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

Robert Peharz, Steven Lang, Antonio Vergari, Karl Stelzner, Alejandro Molina, Martin Trapp, Guy Van den Broeck, Kristian Kersting, and Zoubin Ghahramani. Einsum networks: Fast and scalable learning of tractable probabilistic circuits. In *ICML*, 2020.

Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 689–690. IEEE, 2011.