Learning Sum-Product Networks

Martin Trapp

Probabilistic Machine Learning

- Sum-product networks (SPNs) [Poon2011] are general-purpose probabilistic machine learning models that admit tractable exact probabilistic inference.

- Sum-product networks (SPNs) [Poon2011] are general-purpose probabilistic machine learning models that admit tractable exact probabilistic inference.
- What are *tractable probabilistic models*?

- Sum-product networks (SPNs) [Poon2011] are general-purpose probabilistic machine learning models that admit tractable exact probabilistic inference.
- What are *tractable probabilistic models*?
- A class of queries Q on a class of models M is tractable, iff for any query $q \in Q$ and model $m \in M$ the computational complexity is at most polynomial.

- Sum-product networks (SPNs) [Poon2011] are general-purpose probabilistic machine learning models that admit tractable exact probabilistic inference.
- What are *tractable probabilistic models*?
- A class of queries Q on a class of models M is tractable, iff for any query $q \in Q$ and model $m \in M$ the computational complexity is at most polynomial.
- SPNs admit many probabilistic inference tasks, such as marginalisation, in linear time in their representation size.

Sum-Product Networks

- Sum-product networks (SPNs)[1] is a class of general-purpose probabilistic machine learning models that admit tractable probabilistic inference.
- SPNs are a sub-class of so-called tractable probabilistic models or probabilistic circuits.
- A class of queries Q on a class of models M is tractable, iff for any query $q \in Q$ and model $m \in M$ the computational complexity is at most polynomial.
- SPNs admit many probabilistic inference tasks, such as marginalisation, in linear time in their representation size.

[1] H. Poon & P. Domingos: Sum-product networks: A new deep architecture. In UAI, 2011.

What is a Sum-Product Network?

- Let $\mathbf{X} = \{X_1, \ldots, X_D\}$ be set of D random variables.
- An SPN is a distribution over $\mathbf{X}$ defined as a 4-tuple $\mathcal{S} = (\mathcal{G}, \psi, \boldsymbol{w}, \theta)$.
  - $\mathcal{G}$ is a computational graph.
  - ψ is a so-called scope function.
  - w denotes the set of sum-weights and θ the set of leaf node parameters.

This definition[2] is conceptually different to the original definitions as it disentangles the computational graph and the scope function.

[2] M. Trapp et al.: Bayesian Learning of Sum-Product Networks. In NeurIPS, 2019.

Computational Graph $\mathcal{G}$

$\mathcal{G}$ is a rooted connected directed acyclic graph (DAG), containing: sum (S), product (P) and leaf nodes (L).

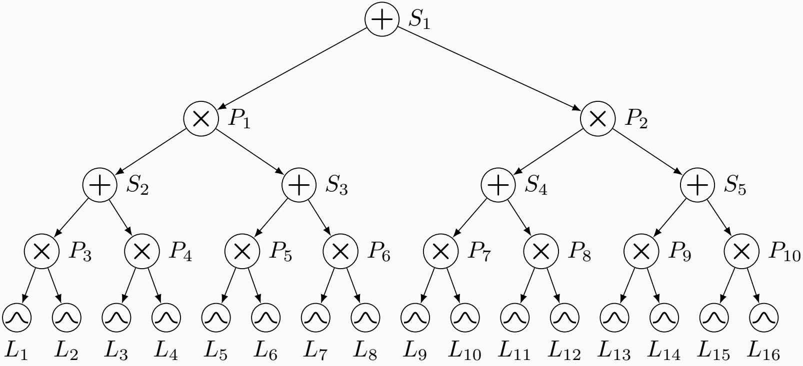
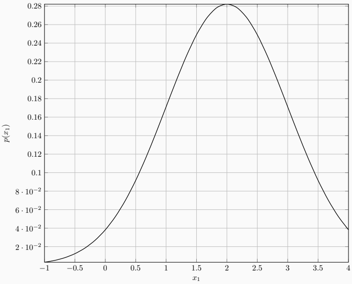


Figure 1: Example of a tree-shaped computational graph.

Leaves L in $\mathcal{G}$

Leaf nodes are input nodes with arbitrary distribution, e.g. Gaussian, Multinomial, variational autoencoder.

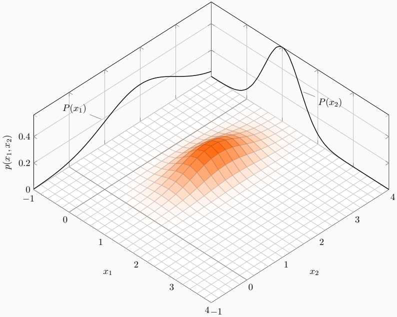
$$\mathsf{L}(x) = p(x \mid \theta_{\mathsf{L}})$$



Product Nodes P in $\mathcal{G}$

Product nodes encode independence assumptions between sets of random variables.

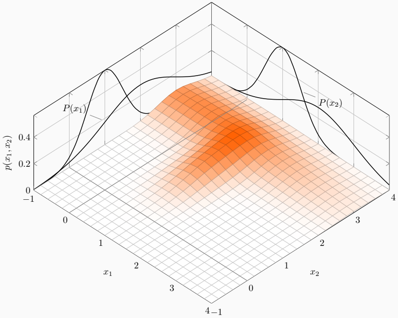
$$\mathsf{P}(x) = \prod_{\mathsf{C} \in \textbf{ch}(\mathsf{P})} \mathsf{C}(x)$$



Sum Nodes S in $\mathcal{G}$

Sum nodes[3] replace independence with conditional independence within the network.

$$\mathsf{S}(x) = \sum_{\mathsf{C} \in \mathbf{ch}(\mathsf{P})} w_{\mathsf{S},\mathsf{C}} \mathsf{C}(x)$$



[3] We assume that $w_{\mathsf{S},\mathsf{C}} \geq 0$.

Scope Function ψ

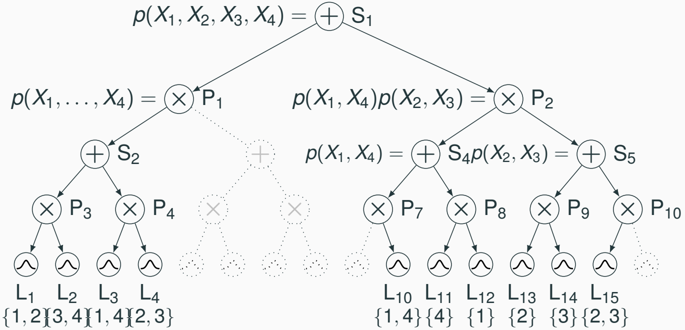
ψ is a function assigning each node N in a sub-set of **X**,[4] and has to fulfil the following properties:

1. If N is the root node, then $\psi(\mathrm{N}) = \mathbf{X}$.
2. If N is a sum or product, then $\psi(\mathrm{N}) = \bigcup_{\mathrm{N}' \in \mathbf{ch}(\mathrm{N})} \psi(\mathrm{N}')$.
3. For each S $\in$ **S** we have $\forall \mathrm{N}, \mathrm{N}' \in \mathbf{ch}(\mathrm{S})\colon \psi(\mathrm{N}) = \psi(\mathrm{N}')$ (*completeness*) [a].
4. For each P $\in$ **P** we have $\forall \mathrm{N}, \mathrm{N}' \in \mathbf{ch}(\mathrm{P})\colon \psi(\mathrm{N}) \cap \psi(\mathrm{N}') = \emptyset$ (*decomposability*).

[a] Complete and decomposable SPNs are referred to as valid SPNs.

[4] This sub-set is often referred to as the scope of a node.

Example SPN $\mathcal{S} = (\mathcal{G}, \psi, \boldsymbol{w}, \theta)$



After applying a scope function ψ on $\mathcal{G}$ we obtain the SPN.
Most structure learners learn both in an entangled way.

Note that we define $\mathsf{L}(x) := 1$ for every x if and only if $\psi(\mathsf{L}) = \emptyset$.