Università degli Studi di Bari
Dipartimento di Informatica

LACAM
Machine Learning

Learning Sum-Product Networks

Nicola Di Mauro **Antonio Vergari**

September 2016

The need for SPN

Sum-Product Networks (SPNs) are a type of probabilistic model[1]

- for Pobabilistic Graphical Models (PGMs) there exist multi-purpose inference tools
  - the computational effort scales unproportional to the complexity of the graph
  - solution: using approximate inference

[1] H. Poon and P. Domingos, *Sum-Product Network: a New Deep Architecture*, UAI 2011

The need for SPNs

Why should you work on SPNs?

- exact tractable inference
- NN for which structure learning is easy

SPNs represent probability distributions and a corresponding exact inference machine for the represented distribution at the same time

Representation

Density estimation

(Different kinds of) Inference

Different kinds of queries:

- $p(\mathbf{X})$ (evidence)
- $p(\mathbf{E}), \mathbf{E} \subset \mathbf{X}$ (marginals)
- $p(\mathbf{Q}|\mathbf{E}), \mathbf{Q}, \mathbf{E} \subset \mathbf{X}, \mathbf{Q} \cap \mathbf{E} = \emptyset$ (conditionals)
- $\arg\max_{\mathbf{q} \sim \mathbf{Q}} p(\mathbf{q}|\mathbf{E})$ (MPE assignment)
- complex queries

Tractable Probabilistic Models

Sum-Product Networks

Scopes

Structural Properties

Inference

Complete evidence

Marginal inference

MPE inference

Interpretation

Interpretation

- probabilistic model
- deep feedforward neural network

Network Polynomials

Arithmetic Circuits

Differences with ACs:

- probabilistic semantics
  - learning
  - sampling
- no shared weights

SPNs as NNs (I)

SPNs are a particular kind of ***labelled constrained*** *and* ***fully probabilistic*** neural networks.

Labelled: each neuron is associated a *scope*
Constrained: completeness and decomposability determine network topology.
Fully probabilistic: each valid sub-SPN is still a valid-SPN.

SPNs provide a direct encoding of the input space into a deep architecture $\rightarrow$ ***visualizing representations*** (back) into the ***input space***.

A. Vergari, N. Di Mauro, and F. Esposito. "Visualizing and Understanding Sum-Product Networks". In: *preprint arXiv* [2016].

SPNs as NNs (II)

A classic MLP hidden layer computes the function:

$$h(\mathbf{x}) = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$$

SPNs can be reframed as *DAGs* of MLPs, each sum layer computing:

$$\mathbf{S}(\mathbf{x}) = \log(\mathbf{W}\mathbf{x})$$

and product layers computing:

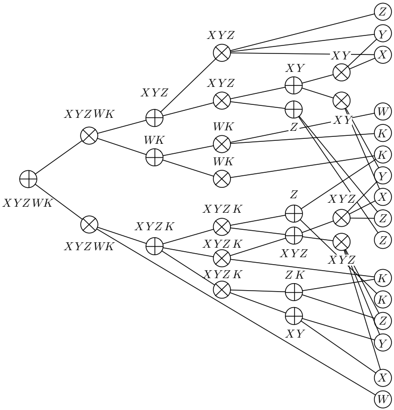
$$\mathbf{S}(\mathbf{x}) = \exp(\mathbf{P}\mathbf{x})$$

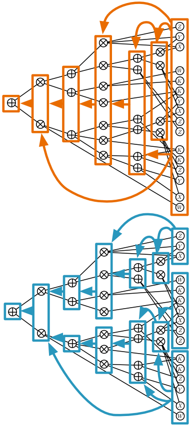
where $\mathbf{W} \in \mathbb{R}_{+}^{s \times r}$ and $\mathbf{P} \in \{0, 1\}^{s \times r}$ are the weight matrices:

$$\mathbf{W}_{(ij)} = \begin{cases} w_{ij} & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases} \qquad \mathbf{P}_{(ij)} = \begin{cases} 1 & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases}$$

A. Vergari, N. Di Mauro, and F. Esposito. "Visualizing and Understanding Sum-Product Networks". In: *preprint arXiv* [2016].

SPNs as NNs (III)





SPNs as NNs (IV): filters

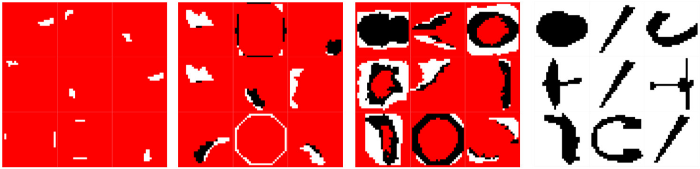
Learned features as images maximizing neuron activations []:

$$\mathbf{x}^{*} = \operatorname*{argmax}_{\mathbf{x}, ||\mathbf{x}|| = \gamma} h_{ij}(\mathbf{x}; \boldsymbol{\theta}).$$

With SPNs, joint solution as an MPE assignment for all nodes (linear time):

$$\mathbf{x}^{*}_{|\mathsf{sc}(n)} = \operatorname*{argmax}_{\mathbf{x}} S_n(\mathbf{x}_{|\mathsf{sc}(n)}; \mathbf{w})$$

.



$\rightarrow$ *scope length* ($|\mathsf{sc}(n)|$) correlates with feature abstraction level

A. Vergari, N. Di Mauro, and F. Esposito. “Visualizing and Understanding Sum-Product Networks”. In: *preprint arXiv* [2016].

SPNs as BNs I

Zhao and Poupart

SPNs as BNs II

Peharz

Myths about SPNs

SPNs are PGMs. false

SPNs are convolutional NNs. false

SPNs

Learning

Structure Learning

Structure matters

Alternatives:

- handcrafted structure, then weight learning []
- random structures, then weight learning []
- learned from data

Why Structure Quality Matters

Tractable inference is guaranteed *if the network size is polynomial* in # vars.

Smaller networks, faster inference (comparing network sizes is better than comparing inference times).

Deeper networks are possibly *more expressively efficient [***Martens2014***,* **Zhao2015** *]*.

Structural simplicity as a bias: overcomplex networks may not generalize well.

Structure quality desiderata: **smaller** but **accurate**, ***deeper*** but not wider, SPNs.

LearnSPN (I)

Build a tree-like SPN by recursively split the data matrix:

- splitting columns into pairs by a greedy ***G Test*** based procedure with threshold ρ:

$$G(X_i, X_j) = 2 \sum_{x_i \sim X_i} \sum_{x_j \sim X_j} c(x_i, x_j) \cdot \log \frac{c(x_i, x_j) \cdot |T|}{c(x_i)c(x_j)}$$

- clustering instances into $|C|$ sets with ***online Hard-EM*** with cluster penalty λ:

$$Pr(\mathbf{X}) = \sum_{C_i \in \mathbf{C}} \prod_{X_j \in \mathbf{X}} Pr(X_j|C_i)Pr(C_i)$$

  weights are estimated as cluster proportions

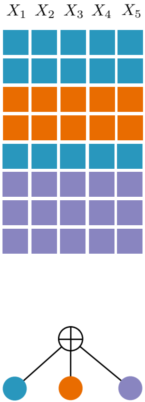
- if there are less than m instances, put a ***naive factorization*** over leaves
- each univariate distribution get ***ML estimation*** smoothed by α

Hyperparameter space: $\{\rho, \lambda, m, \alpha\}$.

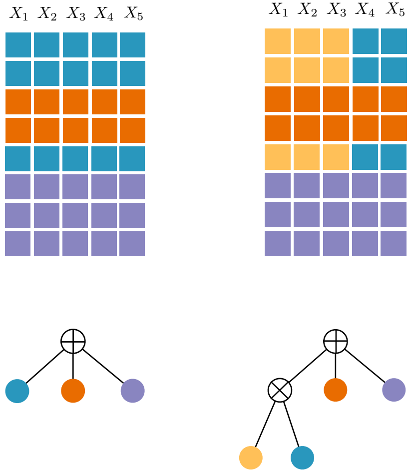
LearnSPN (II)

	X_1	X_2	X_3	X_4	X_5
1					
2					
3					
4					
5					
6					
7					
8					

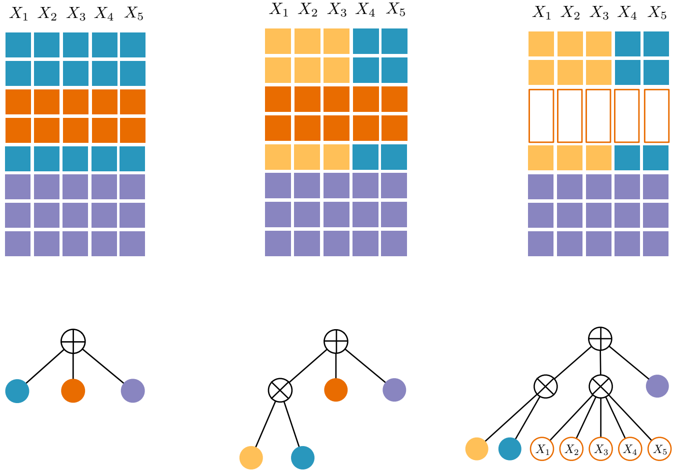
LearnSPN (II)



LearnSPN (II)



LearnSPN (II)



LearnSPN (III)

LearSPN performs two interleaved ***greedy hierarchical*** divisive ***clustering*** processes (co-clutering on the data matrix).

Fast and simple. But both processes never look back and are committed to the choices they take.

Online EM does not need to specify the number of clusters k in advance. But overcomplex structures are learned by exploding the number of sum node children.

Tractable leaf estimation. But naive factorization independence assumptions may be too strong.

ML estimations are effective. But they are not robust to noise, they can overfit the training set easily.

LearnSPN

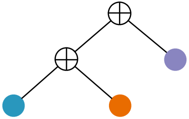
LearnSPN-b

Observation: each clustering process benefits from the other one improvements/highly suffers from other's mistakes.

Idea: slowing down the processes by limiting the number of nodes to split into. SPN-B, variant of LearnSPN that uses EM for mixture modeling with $k = 2$ to cluster rows.
No need for λ anymore.

Objectives:

- not committing to complex structures too early
- same expressive power as LearnSPN
- reducing node out fan increases the depth
- same accuracy, smaller networks



LearnSPN-b: depth VS size

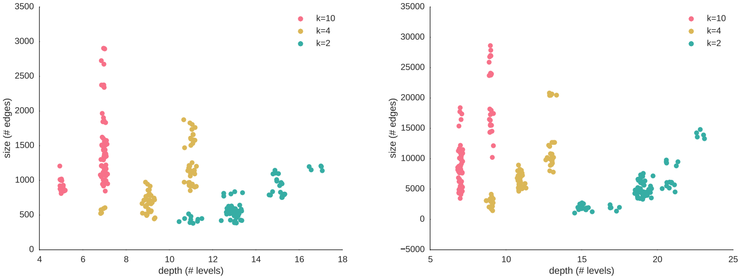


Figure : Comparing network sizes and depths while varying the max number of sum node children splits ($k \in \{10, 4, 2\}$). Each dot is an experiment in the grid search hyperparameter space performed by SPN-B on NLTCS (left) and Plants (right).

New Tendencies in Structure Learning

Pruning and compressing

Parameter Learning

Hard/Soft Parameter Learning

Bayesian Parameter Learning

Parameter Learning VS LearnSPN

Collapsed Variational Inference is useless : D

Representation Learning

Extracting Embeddings

Problem extracting embeddings

Supervised classification

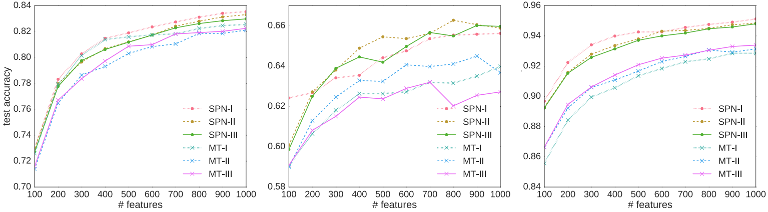
Filtering Embeddings

Filtering embeddings by:

- node type
- scope
- scope length

Random Marginal Queries

Generate embeddings by asking several random queries to a black box density estimator.
Eg. marginals: $e^i_j = P_\theta(\mathbf{Q}_j = \mathbf{x}^i_{\mathbf{Q}_j})$, according to estimator θ where $\mathbf{Q}_j \subseteq \mathbf{X}, j = \ \ldots, k$.



A. Vergari, N. Di Mauro, and F. Esposito. “Visualizing and Understanding Sum-Product Networks”. In: *preprint arXiv* [2016].

Encoding/Decoding Embeddings

MPN as autoencoders[2].

[2] Vergari et al. Encoding and Decoding Representations with Sum-Product Networks, 2016, to appear

Applications

Applications I: computer vision

Applications II: language modeling

Applications III: activity recognition

Applications IV: speech

Trends & What to do next

References

awesome-spn

A curated and structured list of resources about SPNs[3].
`https://github.com/arranger1044/awesome-spn`

[3] Inspired by the SPN page http://spn.cs.washington.edu/ at the Washington University