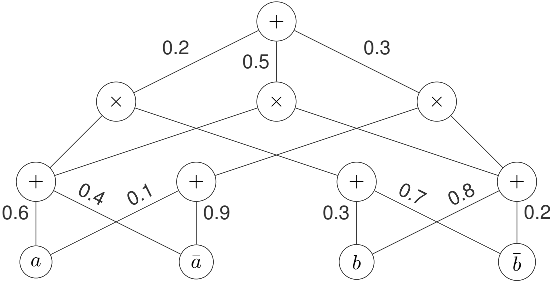
Credal Sum-Product Networks

Denis Mauá
Fabio Cozman
Universidade de São Paulo
Brazil

Diarmaid Conaty
Cassio de Campos
Queen's University Belfast
United Kingdom

ISIPTA 2017

- Sum-Product Networks: sacrifice "interpretability" of Bayesian networks for the sake of computational efficiency; represent computations not interactions (Poon & Domingos 2011).
- Complex mixture distributions represented graphically as an arithmetic circuit.



EXAMPLE (POON AND DOMINGOS 2011)

Learn models from dataset of “faces”; then use it to complete partial face

Distribution $S(X_1, \dots, X_n)$ built by

- an indicator function over a single variable
  - $I(X = 0), I(Y = 1)$ (also written $\neg x, y$),
- a weighted sum of SPNs with same domain and nonnegative weights
  - $S_3(X, Y) = 0.6 \cdot S_1(X, Y) + 0.4 \cdot S_2(X, Y)$,
- a product of SPNs with disjoint domains
  - $S_3(X, Y, Z, W) = S_1(X, Y) \cdot S_2(Z, W)$.

Sum-Product Network

Distribution $S(X_1, \ldots, X_n)$ built by

- an indicator function over a single variable
  - $I(X = 0)$, $I(Y = 1)$ (also written $\neg x, y$),
- a weighted sum of SPNs with same domain and nonnegative weights
  - $S_3(X, Y) = 0.6 \cdot S_1(X, Y) + 0.4 \cdot S_2(X, Y)$,
- a product of SPNs with disjoint domains
  - $S_3(X, Y, Z, W) = S_1(X, Y) \cdot S_2(Z, W)$.

We can assume that weights are normalized: $\sum_i w_i = 1$.

Distribution $S(X_1, \ldots, X_n)$ built by

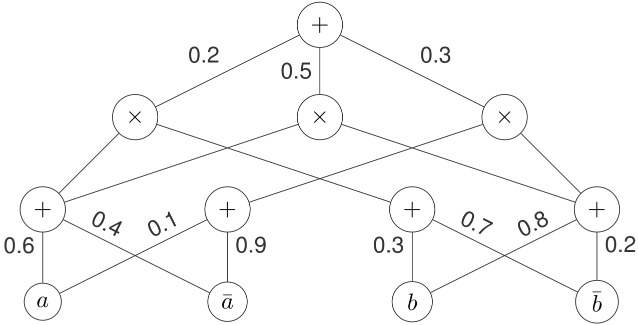
- an indicator function over a single variable
  - $I(X = 0)$, $I(Y = 1)$ (also written $\neg x, y$),
- a weighted sum of SPNs with same domain and nonnegative weights
  - $S_3(X, Y) = 0.6 \cdot S_1(X, Y) + 0.4 \cdot S_2(X, Y)$,
- a product of SPNs with disjoint domains
  - $S_3(X, Y, Z, W) = S_1(X, Y) \cdot S_2(Z, W)$.

We can assume that weights are normalized: $\sum_i w_i = 1$.

Weighted sums have implicit latent variable:
$0.6 \cdot S_1(X, Y) + 0.4 \cdot S_2(X, Y) = \sum_Z P(Z) \cdot P(X, Y|Z)$.

Graphical Representation

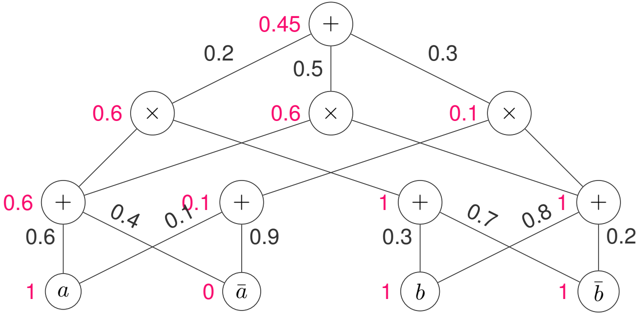
- Rooted directed acyclic graph (directions are implicit below);
- Leaves are indicators;
- Sums and product nodes (edges leaving sum nodes are weighted).



Evaluation (Inference)

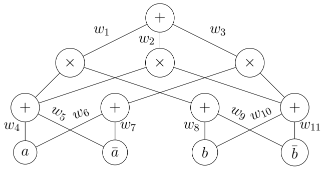
- Propagate values bottom-up:

$P(A = 1) =$



Note: takes linear time in the size of circuit!

- Robustify SPNs by allowing weights to vary inside sets (for instance, towards sensitivity analisys on SPN's inference).
- New class of tractable imprecise graphical models.



$$
\left.\begin{array}{cc}
(w_1, w_2, w_3) \in \mathrm{CH}(& [0.28, 0.45, 0.27], \\
[0.18, 0.55, .27], & [0.18, 0.45, 0.37]), \\
0.54 \leq w_4 \leq 0.64, & 0.36 \leq w_5 \leq 0.46, \\
0.09 \leq w_6 \leq 0.19, & 0.81 \leq w_7 \leq 0.91, \\
0.27 \leq w_8 \leq 0.37, & 0.63 \leq w_9 \leq 0.73, \\
0.72 \leq w_{10} \leq 0.82, & 0.18 \leq w_{11} \leq 0.28, \\
w_4 + w_5 = 1, & w_6 + w_7 = 1, \\
w_8 + w_9 = 1, & w_{10} + w_{11} = 1
\end{array}\right\}
$$

- To compute upper and lower unconditional probabilities.
- To compute upper and lower (conditional) expectations.
- To perform credal classification.
- To analyze robustness of SPNs.

Consider a CSPN $\{S_w : w \in \mathcal{C}\}$, where $\mathcal{C}$ is the Cartesian product of finitely-generated polytopes $\mathcal{C}_i$, one for each sum node i.

THEOREM

Computing $\min_w S_w(x)$ *and* $\max_w S_w(x)$ *takes polynomial time in the size of circuit and representation of sets* $\mathcal{C}_i$*.*

- Similar to evaluation of SPNs, except that sum nodes need to solve LP.

Given CSPN $\{S_w : w\}$, compute $\min_w \sum_{q,e} f(q,e) \frac{S_w(q,e)}{\sum_{q',e} S_w(q',e)}$.

Given CSPN $\{S_w : w\}$, compute $\min_w \sum_{q,e} f(q,e) \frac{S_w(q,e)}{\sum_{q',e} S_w(q',e)}$.

THEOREM

Assuming that f is encoded succinctly, computing lower conditional expectation is NP-hard.

Given CSPN $\{S_w : w\}$, compute $\min_w \sum_{q,e} f(q,e) \frac{S_w(q,e)}{\sum_{q',e} S_w(q',e)}$.

Theorem

Assuming that f is encoded succinctly, computing lower conditional expectation is NP-hard.

Theorem

Computing lower conditional expectations of a univariate f takes at most polynomial time when each internal node has at most one parent.

Note: Most structure learning algorithms generate SPNs of the above form!

Given configurations c', c'' of variables C and evidence e decide:

$$\min_{w} \left(S_w(c', e) - S_w(c'', e) \right) > 0.$$

THEOREM

Credal classification is coNP-complete.

Given configurations c', c'' of variables C and evidence e decide:

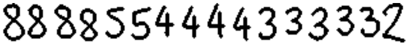
$$\min_{w} \big(S_w(c', e) - S_w(c'', e) \big) > 0.$$

THEOREM

Credal classification is coNP-complete.

THEOREM

Credal classification with a single class variable can be done in polynomial time when each internal node has at most one parent.



- Handwritten digit recognition (70 handwritten 20×30 images per digit).
- We learn and check accuracy of an SPN using 50% - 50% and 20% - 80% train-test splits (ramdomly multiple times).
- Robustness index: maximum ϵ s.t. locally ϵ-contaminating weights of SPN does not change classification (that is, single maximal=single e-admissible class).
- Compared against threshold-based robustness (when best - second best probability is below a threshold).

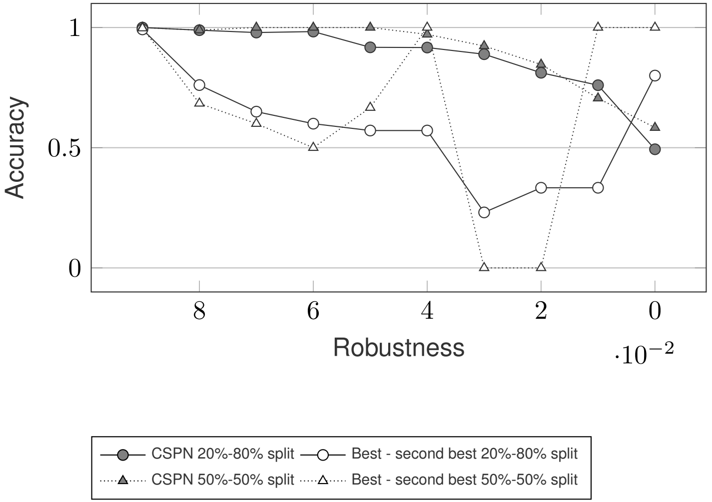


FIGURE: Average classification accuracy for instance below the given robustness (x-axis values are multiplied by 20 to be visually compatible with the probabilities).

Robustness Measure	CSPN		Best - second best	
	Correct	Wrong	Correct	Wrong
median	0.0363	0.0029	0.0909	0.0880
maximum	0.1524	0.0199	0.3333	0.3333
mean	0.0369	0.0043	0.0976	0.1042

TABLE: Robustness values for 50%/50% data split. Overall classification accuracy of 99.31%.

Summing Up!

- Sum-Product Networks offer a recently developed class of probabilistic graphical models with linear time inference.
  - Very promising results in “deep learning”: image completion, image classification from pixels, representation learning, etc.
- Credal Sum-Product Networks extend SPNs to imprecise setting:
  - Unconditional inference still takes polynomial time.
  - Conditional expectation is NP-hard; **very useful subclass admits polytime inference**.
  - Credal classification is also coNP-hard; again **with very useful tractable subclass**.
  - Encouraging preliminary results on handwritten digit recognition task (code available upon request).