



# Aprendizado Automático de Sum-Product Networks

Renato Lui Geh, Orientador: Denis Deratani Mauá

Instituto de Matemática e Estatística, Universidade de São Paulo - MAC0215 Atividade Curricular em Pesquisa

## Motivação

Uma distribuição de probabilidades pode ser representada por uma função multilinear nas variáveis de uma distribuição com um número potencialmente exponencial de termos (ou seja, não compacta) como

$$P(x_1, \dots, x_n) = \sum_{\alpha} c_{\alpha} \prod_{i \in \alpha} x_i.$$

O objetivo em Modelos Gráficos Probabilísticos (PGM) é computar inferência, ou seja, deseja-se encontrar a probabilidade

$$P(x_q | x_{e_1}, \dots, x_{e_m}) = \frac{P(x_q x_{e_1} \dots x_{e_m})}{P(x_{e_1} \dots x_{e_m})}, \text{ onde } x_q \text{ é a query e } x_{e_1}, \dots, x_{e_m} \text{ é a evidência.}$$

No entanto, inferência na maioria dos PGMs é intratável e, apesar de existirem modelos onde a inferência é, de fato, tratável, e serem representações compactas de distribuições, elas são limitadas em sua flexibilidade.

Em 2011[PD11], Pedro Domingos e Hoifung Poon introduziram um novo tipo de modelo probabilístico que representa eficientemente uma função multilinear através de um digrafo acíclico enraizado (DAG), cuja inferência é sempre tratável e ainda assim é mais flexível que muitos outros modelos. Por meio de experimentos também comprovou-se que tanto inferência quanto aprendizado foram mais rápidos e precisos que outras redes profundas.

O objetivo desse estudo é aprender a definição, estrutura e propriedades de Sum-Product Networks e em seguida estudar os vários tipos de aprendizado que podemos efetuar neste modelo.

## Sum-Product Networks

Uma Sum-Product Network (SPN) tem definição recursiva. Seja  $S$  uma SPN:

- ▶ Uma distribuição monovariável  $P(X_i)$  é uma SPN.
- ▶ A soma  $w_i S_i(X_{\alpha}) + w_j S_j(X_{\beta})$  com pesos  $w_j, w_i \geq 0$  é uma SPN. (1)
- ▶ O produto  $S_i(X_{\alpha}) \cdot S_j(X_{\beta})$  é uma SPN. (2)

Podemos representar uma SPN com variáveis  $x_1, \dots, x_d$  como um digrafo acíclico enraizado (DAG) cujas folhas são indicadores  $x_1, \dots, x_d$  e  $\bar{x}_1, \dots, \bar{x}_d$  e cujos nós internos são nós somas ou produtos. Toda aresta  $ij$  onde  $i$  tem origem em um nó soma tem um peso  $w_{ij} \geq 0$  associado. O valor de um nó  $i$  é  $v_i$ . O valor de um nó soma  $i$  é  $\sum_{j \in Ch(i)} w_{ij} v_j$ . O valor de um nó produto  $i$  é  $\prod_{j \in Ch(i)} v_j$ .  $Ch(i)$  é o conjunto de nós filhos de  $i$ . O valor de um nó folha é o valor do indicador. O valor de uma SPN  $S$  é o valor de sua raiz.

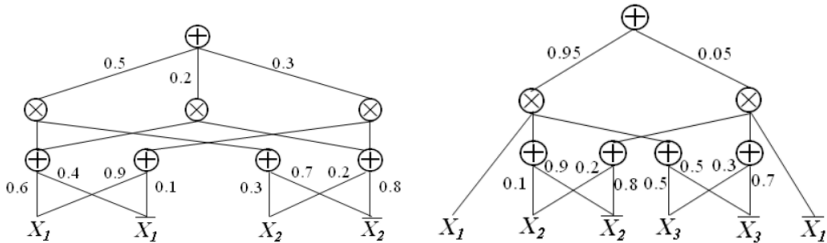


Figura : À esquerda uma SPN implementando uma naive Bayes mixture model. À direita uma SPN implementando uma junction tree. Fonte: Poon e Domingos[PD11].

**Definição** Uma SPN  $S$  é válida sse  $\exists P : S(x) = P(x), \forall x$ .

**Definição** Uma SPN  $S$  é completa sse  $\alpha = \beta$  em (1).

**Definição** Uma SPN  $S$  é consistente sse  $\alpha \cap \beta = \emptyset$  em (2).

**Teorema** Uma SPN  $S$  é válida se  $S$  é completa e consistente.

SPNs válidas são desejáveis pois computam  $P(x_1, \dots, x_n)$  em tempo linear em seu tamanho, além de completude e consistência permitirem que a inferência da SPN seja garantidamente eficiente.

## Aprendizado

Pode-se dividir aprendizado de SPNs em duas classes:

- 1 A partir de um DAG da SPN pré-definido, aprendemos os pesos do digrafo.
- 2 Ambos DAG e pesos são desconhecidos e são aprendidos.

O algoritmo proposto em [PD11] segue a primeira classe e é mostrado na seção seguinte. A partir de uma SPN densa e válida podemos aprender os pesos por Gradient Descent ou Expectation-Maximization (EM).

Gens e Domingos[GD13] propõem um outro método de aprendizado que explora a expressividade da SPN aprendendo-se não só os pesos como o próprio DAG. Para aprender o digrafo pode-se maximizar o estimador de máxima verossimilhança  $\max_S P(X^1, \dots, X^N | S)$ , onde  $X^1, \dots, X^N$  são os conjuntos de dados.

## Algoritmo de Aprendizado de Pesos

**Input:** Conjunto  $D$  de instâncias sobre variáveis  $X$ .

**Output:** Uma SPN com estrutura e parâmetros construídos por aprendizado.

```
/* Cria uma SPN inicial que seja válida. */
S ← GenerateDenseSPN(X);
InitializeWeights(S);
repeat
  forall the d ∈ D do
    /* Atualiza pesos por Gradient Descent ou EM. */
    UpdateWeights(S, Inference(S, d));
  end
until convergência;
/* Apara arestas com peso wij = 0 e nós não-raiz sem pais. */
S ← PruneZeroWeights(S);
return S
```

## Experimentos

Os experimentos mostrados a seguir foram extraídos a partir da implementação do algoritmo mostrado na seção anterior e mostram os resultados do código [DP] implementado por Domingos e Poon e citados em [PD11].

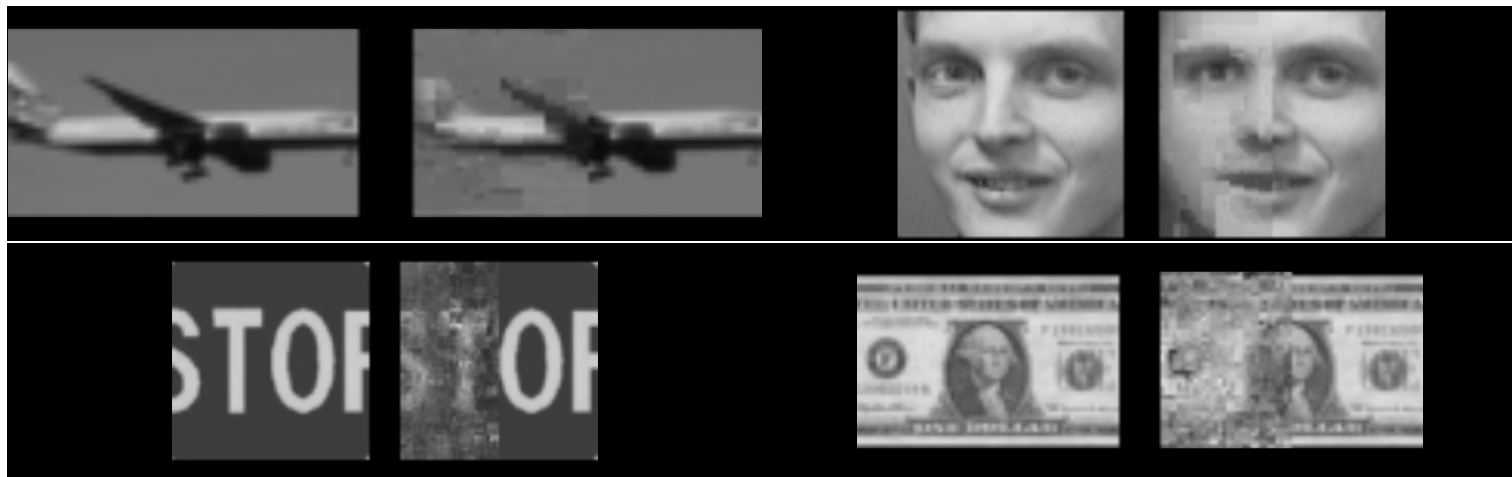


Figura : A saída do algoritmo consiste na compleção do lado esquerdo das imagens a partir de um conjunto de treino. Para cada par de imagens, a imagem da esquerda é a original, enquanto que a direita tem a metade esquerda completada pela SPN e a outra metade igual a da original como evidência.

Arquitetura	Rostos	Motos	Carros
SPN	99%	99%	98%
CDBN	95%	81%	87%

Tabela : Taxa média de acertos em uma comparação entre SPNs e CDBNs (Convolutional Deep Belief Networks) em classificação (reconhecimento) de imagens. SPNs obtiveram resultados quase perfeitos em reconhecimento.

Pode-se ver que os resultados das SPNs são muito promissores e, dado que o algoritmo produzido por Domingos e Poon não toma muita vantagem da expressividade da estrutura local de SPNs, é fácil notar que ainda há muito espaço para melhorias.

## Trabalhos futuros

Pretende-se estudar a implementação do método de aprendizado proposto por Poon e Domingos[PD11], realizar outros experimentos com este algoritmo e explorar mais a fundo as propriedades de uma SPN.

Em seguida planeja-se estudar outros tipos de aprendizado em SPNs, principalmente métodos que estejam contidos na classe 2 de aprendizado e portanto tomem vantagem da estrutura local de SPNs, como o introduzido por Gens e Domingos[GD13], buscas gulosas e clustering por Dennis e Ventura[DV12, DV15] e Non-Parametric Bayesian Sum-Product Networks[LWZ14].

## Referências

- Pedro Domingos and Hoifung Poon. Sum-product networks: A new deep architecture (code). URL: <http://spn.cs.washington.edu/spn/>.
- Aaron Dennis and Dan Ventura. Learning the architecture of sum-product networks using clustering on variables. *Advances in Neural Information Processing Systems*, 25, 2012.
- Aaron Dennis and Dan Ventura. Greedy structure search for sum-product networks. *International Joint Conference on Artificial Intelligence*, 24, 2015.
- Robert Gens and Pedro Domingos. Learning the structure of sum-product networks. *International Conference on Machine Learning*, 30, 2013.
- Sang-Woo Lee, Christopher Watkins, and Byoung-Tak Zhang. Non-parametric bayesian sum-product networks. *Workshop on Learning Tractable Probabilistic Models*, 2014.
- Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. *Uncertainty in Artificial Intelligence*, 27, 2011.