Automatic Learning of Sum-Product Networks

Renato Lui Geh (student), Denis Deratani Mauá (advisor)
Institute of Mathematics and Statistics, University of São Paulo
{renatolg,ddm}@ime.usp.br

Introduction and Methods

Sum-product networks (SPNs) are deep probabilistic graphical models capable of representing tractable probability distributions with a great number of variables. In the last decade, SPNs have achieved impressive results in various fields, such as protein folding, signal modelling, image classification and completion, activity recognition and natural language. Graphically, SPNs can be seen as DAGs whose leaves are tractable univariate probability distributions and internal nodes are either weighted sums or products. Computing exact inference in SPNs is done in linear time to the number of the graph's edges.

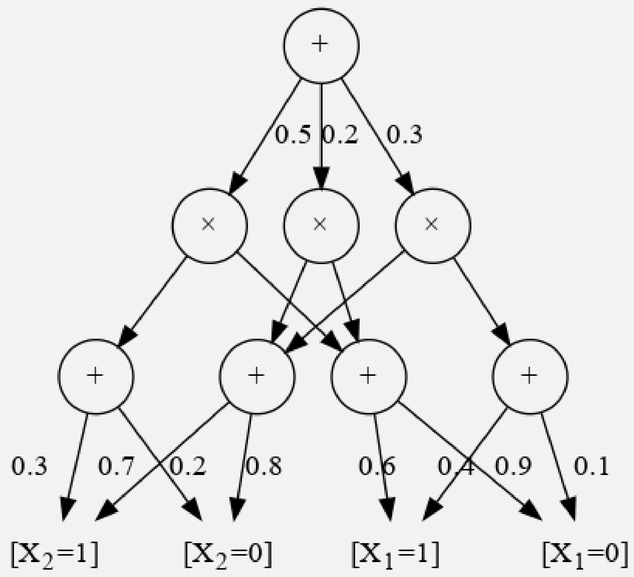

Figure 1: An SPN over variables X_1 and X_2.

Despite these promising results, at present there are very few SPN inference and learning codebases publicly available, and no well maintained, documented and updated SPN library. In addition, there has been no comparative study on different SPN learning methods yet. This project sought to develop a free, open-source inference and learning SPN library, and to compare three different state-of-the-art SPN learning algorithms in the domain of image classification and completion.

We implemented three learning algorithms: the **Poon-Domingos** architecture [3], **Dennis-Ventura** structure algorithm [1] and **Gens-Domingos** schema [2] as part of the GoSPN library (available at `https://github.com/RenatoGeh/gospn/`). We used four datasets for image classification and completion. DigitsX and MNIST are handwritten digits datasets, Caltech-101 is a categorical object dataset and Olivetti is a faces recognition dataset. For Caltech-101 we used only three categories (bikes, cars and faces) due to memory and time constraints.

Development

During training, the Poon-Domingos algorithm struggled with the time and memory constraints established, often exceding either or both. The Gens-Domingos and Dennis-Ventura algorithms had similar running times for training, with both taking less than eight hours in all datasets. Tests were done in an Intel i7–4500U 1.8GHz with 16GB RAM, and were limited to 24 hours of training. Inference was very fast, taking about a second to compute the approximate MAP states.

Training was done without any additional feature extraction. SPNs were trained based solely on pixel values. For the Poon-Domingos and Dennis-Ventura structures, each pixel was represented by a mixture of gaussians with g components. Each pixel was then divided into g quantiles, with each component's mean and standard deviation set to that of its corresponding quantile. The Gens-Domingos algorithm represented each pixel as a discrete multinomial distribution.

Classification was done by taking a percentage p of the dataset as training set, and $1-p$ as test set. Training and test sets were balanced, i.e. every class had the same number of instances each. Predicting labels was done by finding the MPE of the classification variable through approximate MAP state extraction. We speculate that better results could be achieved by computing the exact probability of evidence of each possible valuation instead of the MAP way, albeit with longer inference times.

For image completion, half of the image was given as evidence to the model. The other half was then generated by the SPN by querying for the most probable variable valuations given the evidence.

Results

Due to the issues mentioned earlier, the Poon-Domingos model either did not complete training or had unsatisfactory results. In this poster, we show only results achieved by the Dennis-Ventura (DV) and Gens-Domingos (GD) algorithms.

$p=0.1$	DV	GD
DigitsX	92.85	91.27
Caltech	78.58	77.40
Olivetti	83.78	2.50
MNIST	77.85	81.55

$p=0.2$	DV	GD
DigitsX	98.57	96.78
Caltech	78.49	85.00
Olivetti	74.88	2.50
MNIST	77.85	81.55

$p=0.3$	DV	GD
DigitsX	99.18	96.93
Caltech	79.88	80.28
Olivetti	89.93	84.28
MNIST	77.85	81.55

$p=0.4$	DV	GD
DigitsX	98.81	98.09
Caltech	79.88	86.11
Olivetti	89.93	91.25
MNIST	77.85	81.55

$p=0.5$	DV	GD
DigitsX	99.42	97.14
Caltech	81.38	88.66
Olivetti	89.93	95.50
MNIST	77.85	81.55

$p=0.6$	DV	GD
DigitsX	99.28	97.85
Caltech	81.35	90.00
Olivetti	97.50	98.75
MNIST	77.85	81.55

$p=0.7$	DV	GD
DigitsX	98.57	97.61
Caltech	75.45	92.22
Olivetti	92.89	81.93
MNIST	77.85	81.55

$p=0.8$	DV	GD
DigitsX	93.33	92.66
Caltech	74.78	90.00
Olivetti	50.00	81.59
MNIST	77.85	81.55

$p=0.9$	DV	GD
DigitsX	88.75	86.25
Caltech	75.75	84.84
Olivetti	60.93	100.0
MNIST	77.85	81.55

Table 1: Classification accuracy (in %).

For small p values, the Dennis-Ventura algorithm gave the best results. It also got better scores when the classification variable had many categories. Nonetheless, the Gens-Domingos algorithm still yielded better results overall.

Image completion was done on the Olivetti dataset. The gray half of the image was given as evidence, and the other half in green is the completion generated by the model. The completion on the left was done on an SPN trained with the Gens-Domingos algorithm. The one on the right was generated by a Dennis-Ventura trained SPN.

Figure 2: Image completion on the Olivetti dataset.

Though completion yielded empirically good results in most cases, there were instances where the models had trouble completing key features, such as eyes, nose and mouth, when these were partially or completely occluded. Experiments showed that the SPNs trained proved capable of completing the face's shape and hair, but sometimes had trouble with finer details, such as glasses or facial hair.

Figure 3: Absence or deformation of key features caused by completion.

Conclusions

We achieved good classification results without any feature extraction on different applications, such as handwritten digit classification, object identification and face recognition. On the completion task, the algorithms were able to identify key features, such as nose, eyes and mouth on the Olivetti dataset. Both learning and inference code were documented and made available as part of the free open-source GoSPN library.

Acknowledgements

We would like to thank Diarmaid Conaty and Cassio P. de Campos, both from Queen's University Belfast, for contributing with bug fixes, suggestions and code implementation for the GoSPN library. This project received financial support from CNPq grant PIBIC 800585/2016–0.

References

[1] Aaron Dennis and Dan Ventura. "Learning the Architecture of Sum-Product Networks Using Clustering on Variables". In: *Advances in Neural Information Processing Systems* 25 (2012).

[2] Robert Gens and Pedro Domingos. "Learning the Structure of Sum-Product Networks". In: *International Conference on Machine Learning* 30 (2013).

[3] Hoifung Poon and Pedro Domingos. "Sum-Product Networks: A New Deep Architecture". In: *Uncertainty in Artificial Intelligence* 27 (2011).