# VISUALIZING GENERATIVE SUM-PRODUCT NETWORKS ON IMAGE RECONSTRUCTION

*Renato L. Geh*

University of São Paulo
Institute of Mathematics and Statistics
Rua do Matão, 1010 - São Paulo, SP, Brazil, 05508-090

## ABSTRACT

Sum-Product Networks (SPNs) are fairly recent deep tractable probabilistic graphical models that are able to answer exact queries in linear time. Although there have been many advancements in practical problems, there is an absence in literature of visualizations on how SPNs represent learned data. In this paper we show how two structure learning algorithms can heavily impact on how SPNs treat data, particularly in the domain of image reconstruction. We show a coloring technique to visualize sum and product nodes through their scopes. We then apply this technique to generative SPNs learned from two distinct learning methods.

***Index Terms***— Sum-product networks, probabilistic graphical models, visualization, image reconstruction

## 1. INTRODUCTION

Image reconstruction is the task of accurately predicting, guessing and completing missing elements from an image. Density estimators that model a joint probability distribution can achieve this by learning the features of similar images and finding the valuation that most adequately fits the incomplete image. However, classical density estimators, such as Probabilistic Graphical Models (PGMs), suffer from exact inference intractability in the general case. This leads to approximate prediction and representation, as learning in PGMs often requires the use of inference as a subroutine.

Sum-Product Networks (SPNs) [1] are fairly recent tractable PGMs capable of representing distributions as a deep network of sums and products. Most importantly, SPNs are capable of exact inference in time linear to its graph's edges. There have been many advances on SPNs in the image domain, such as image classification and reconstruction [2, 3, 4], image segmentation [5] and activity recognition [6, 7, 8]. However, there have been little effort [9] so far to explore SPNs' semantics and representation power.

In this paper, we provide visualizations on SPNs learned from two structure learning algorithms. We propose a technique to perform this task. This technique relies on a couple of properties SPNs must follow in order to correctly represent a probability distribution, and are highly dependent on the graph's structure. We first give a short background review of SPNs, relevant properties and scope definition. We follow this with an explanation on how we achieved the visualizations shown in this article. Finally, we show results and provide a conclusion of our findings.

## 2. BACKGROUND

An SPN can be seen as a DAG with restrictions with respect to its node types and weighted edges. Let $n$ be a graph node. The set of nodes $\text{Ch}(n)$ is the set of children of $n$. A weighted edge $i \to j$ is denoted by $w_{i,j}$.

**Definition 1.** *A sum-product network (SPN) is a directed acyclic graph. A node $n$ of an SPN can either be a:*

1. *sum, where its value is given by $v_n = \sum_{j \in Ch(n)} w_{n,j} v_j$;*
2. *product, where its value is given by $v_n = \prod_{j \in Ch(n)} v_j$;*
3. *probability distribution, whose value is its probability of evidence.*

In this paper, we assume that all SPN leaves (i.e. node type 3) are tractable univariate distributions, that is, computing its mode or partition function takes constant time. The scope of a node, denoted by $\text{Sc}(n)$, is the union set of the scope of its children.

**Definition 2** (Completeness). *An SPN is complete iff every child of a sum node has the same scope as its siblings.*

**Definition 3** (Decomposability). *An SPN is decomposable iff every child of a product node has disjoint scope with its siblings.*

A complete and decomposable SPN correctly computes the probability of evidence of the modeled distribution. An SPN that correctly represents a probability distribution is said to be valid. In fact, completeness and consistency (i.e. no two children of an SPN node have contradicting variable values) is sufficient (though not necessary) for validity [1]. However,

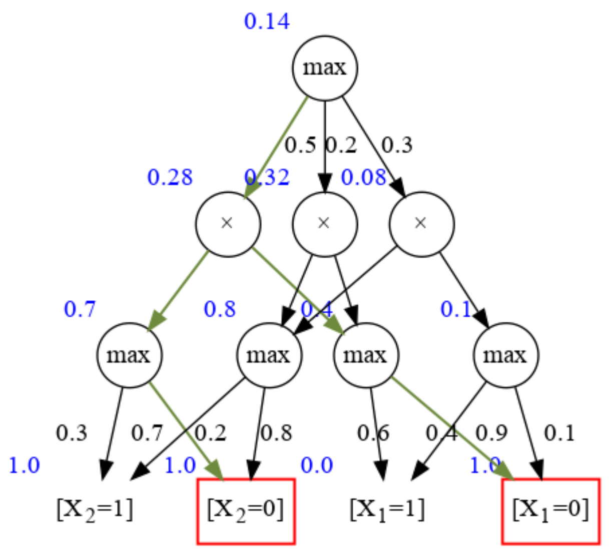

**Fig. 1**. Finding the MPE of an SPN given $\mathbf{X} = \{X_1 = 0\}$.

learning decomposable SPNs is easier, and it has been shown that decomposability is as expressive as consistency [10].

Let $\mathbf{X} = \{X_1 = x_1, X_2, = x_2, \ldots, X_n = x_n\}$ be a valuation and $S$ an SPN. The value of $S$ is the value of its root, and is denoted by $S(\mathbf{X})$. Inference in SPNs is done through a bottom-up evaluation. The value of a leaf node $n$ is the probability of $\mathbf{X}$. If $\text{Sc}(n) \not\subset \mathbf{X}$, then $n$'s value is the distribution's mode.

Finding the $\arg\max_{\mathbf{x}} S(\mathbf{X} = \mathbf{x})$, also called the Most Probable Explanation (MPE), of an SPN has been shown to be NP-hard [10, 11, 12]. Image reconstruction can be seen as an application of MPE, where each variable is a pixel, and values are pixel colors. Finding the MPE, and thus the reconstruction of an image given some initial evidence consists of finding the pixel values that are most likely to fit the model. Given that finding the exact valuation that maximizes the model is hard, we instead use an approximate method proposed in [1] called the Max-Product algorithm.

The Max-Product algorithm consists of replacing the original SPN with a Max-Product Network (MPN). An MPN of an SPN is simply the SPN with its sum nodes replaced with max nodes. The value of a max node is the maximum weighted child. Finding an MPE approximation on an MPN is done through a bottom-up evaluation similar to an SPN. Once all nodes have been computed, a top-down traversal is done, finding the max paths in the graph by choosing only the max edge in a max node and traversing all edges in product nodes, as shown in Figure 1.

## 3. VISUALIZING SPNS

Visualization in SPNs can be done through an analysis of the SPN's scope and structure. The definition of completeness

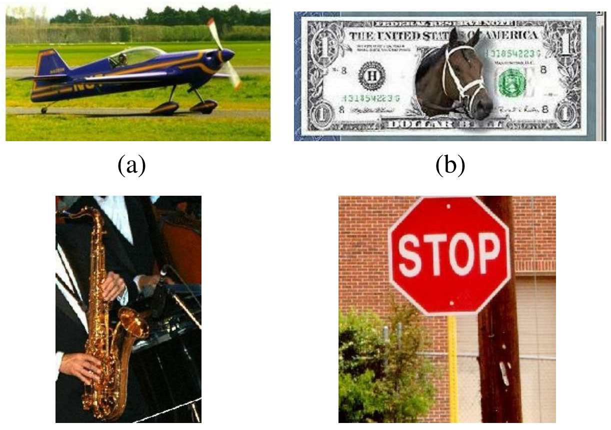

**Fig. 2**. Samples from the Caltech-101 [13] dataset. Categories are, from left to right, top to bottom, (a) airplane, (b) dollar bill, (c) saxophone and (d) stop sign.

lends itself naturally to an interpretation of sum nodes as layers of mixture models. A possible intuition for this interpretation is that sum nodes model latent variables in charge of explaining similar interactions between variables. Decomposability, on the other hand, models independence between sets of variables.

This kind of semantics allows for a very intuitive representation when dealing with images. In the image domain, a product can be thought of as a hidden variable explaining different regions of pixels of an image, with a region being a particular meaningful portion of the image. For instance, in Figure 2 (d), the sign itself could be a region, the bush on the left another, the window, wall and pole the remaining three others. Sum nodes, on the other hand, represent the regions

**Algorithm 1** VisualizeSPN

```
Input  An SPN $S$ and scope threshold $k$
 1: ComputeScope($S$)
 2: for each node $n$ in $S$ in breadth first search order do
 3:     if $\text{Sc}(n) < k$ then
 4:         Skip current search branch
 5:     if $n$ is a product then
 6:         Let $I$ be an image
 7:         for each node $c \in \text{Ch}(n)$ do
 8:             Colorize $\text{Sc}(c)$ onto $I$ with unused color
 9:             if using variation 1 then
10:                 if $c$ is sum and $|\text{Sc}(c)| > k$ then
11:                     Find $c$'s MPE and write to file
12:         Write $I$ to file
13:     else if $n$ is a sum and using variation 2 then
14:         Find $n$'s 3 highest valued weighted children
15:         Find the MPE for each of them and write to file
```

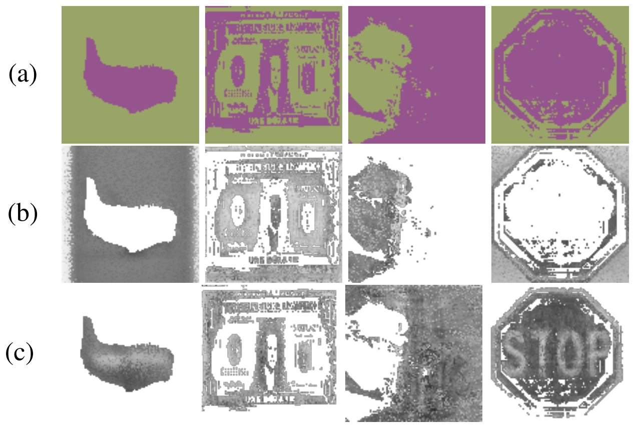

**Fig. 3**. Visualizations at layer 1 (i.e. root's children) for D-SPNs. Row (a) shows product node colorizations. Row (b) and (c) show MPE values for row (a)'s child nodes. Columns from left to right belong to airplanes, dollar bills, saxophones and stop signs.

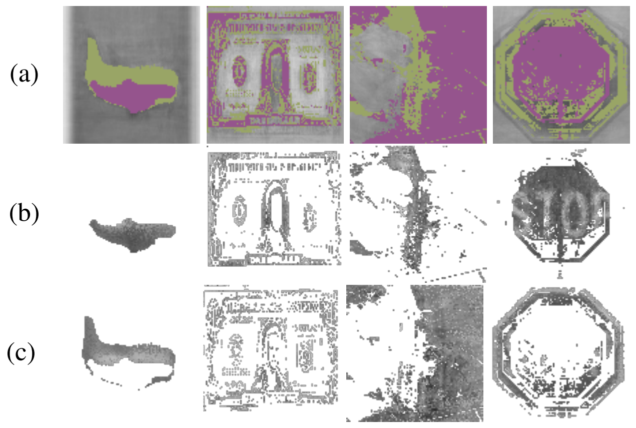

**Fig. 4**. Visualizations at layer 25 for D-SPNs. The deeper the nodes, the more restricted the scope. Colored parts in row (a) indicate to which child the pixels belong to, whereas grayscale values do not belong to the node's scope, and are mean images of all dataset samples.

themselves. For example, another stop sign image may portray a car in place of a bush. Sums model this difference by assigning similar samples under the same child, so that each child models a different object or texture.

We build our visualization technique on top of this interpretation of SPNs. Our method computes the scope of each node. At each product node, we color different regions different colors to distinct between them. For sum nodes, we take two approaches. Variation 2 takes every sum node's MPE and writes it to a file, showing what the most probable objects and textures the node captures, whilst variation 1 is restricted to only sum nodes with at least one product as parent, as sums with only sum nodes as parents are not as expressive and often model only minor changes. We also skip any nodes whose scopes are too small.

By traversing the graph in a breadth-first search manner, we are able to restrict scopes which are too small. Once we identify that the scope of a node has fallen bellow a threshold, we no longer need to keep iterating over remaining descendants, as a complete and decomposable SPN guarantees that every node below it has at most its size. Nodes whose scopes are too small are not as interesting for visualization, as they do not have as much value in semantics, and are often too low-level.

## 4. RESULTS

We applied VisualizeSPN[1] on two SPNs learned with different learning algorithms. One with LearnSPN [2], which we will call G-SPNs, and the other with Dennis and Ventura's clustering algorithm [3], refered here as D-SPNs. The former is implemented with $k$-means for clustering, with a chosen value of $k = 4$, and G-test for variable independence. Leaves are multinomial distributions on the pixels. The latter contains four sums per region and have gaussian mixtures of four gaussians as leaves. We applied variation 1 to D-SPNs and variaton 2 to G-SPNs.

SPNs were learned on the Caltech-101 [13] and Olivetti Faces [14] datasets. The former was reduced to only four categories: airplanes, dollar bills, saxophones and stop signs; and were resized and converted to $100 \times 100$ grayscale images. The Olivetti dataset is already in grayscale, and images were kept in their original $46 \times 56$ dimension.

VisualizeSPN was able to provide clean visualizations on how each learning algorithm segmented data. Figure 3 shows a top-level visualization on D-SPNs, where the top row shows a product node at layer 1. Different colors represent different

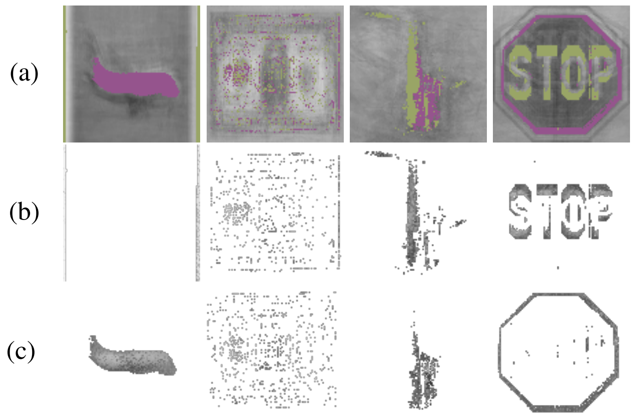

**Fig. 5**. Visualizations at deeper layers for D-SPNs. At lower levels, scopes are more restricted to details. The outline of the saxophone is visible on the third column.

[1]Code available at `https://github.com/RenatoGeh/visualize`

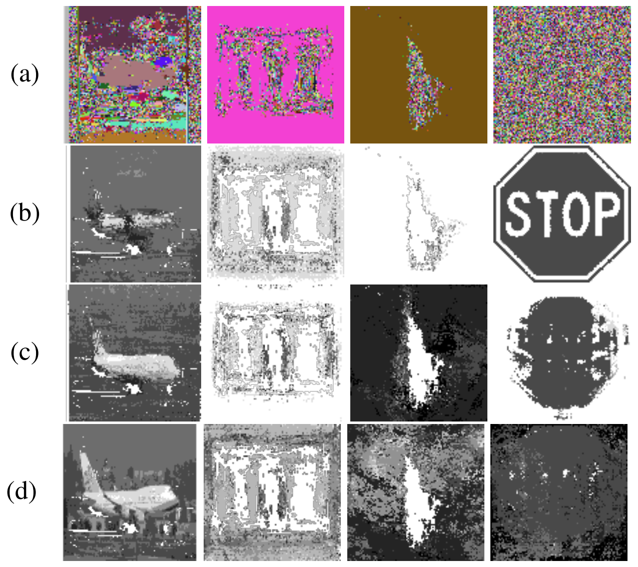

**Fig. 6**. Visualizations for G-SPNs. Row (a) corresponds to the segmentation of a product node's scope. Different colors mean different children. Rows (b), (c) and (d) show the top 3 image reconstructions on (a)'s sum children.

child scopes. The following two rows correspond to the MPE values (i.e. the most probable image reconstructions) on each sum node child. Since the SPN is decomposable, the union of the two children's scope should match the entire parent node's scope. Figure 4 and Figure 5 show deeper layers in the same SPNs. These deeper layers often attempt to model more delicate differences between images, a natural consequence of narrowing the node's scope.

We found that, under a low quantity of samples, G-SPNs were less deep than D-SPNs. We speculate this is due to the approximate nature of the G-test variable independence algorithm. Figure 6 shows how fragmented scopes became, which indicate G-SPNs are much wider and shallower than D-SPNs.

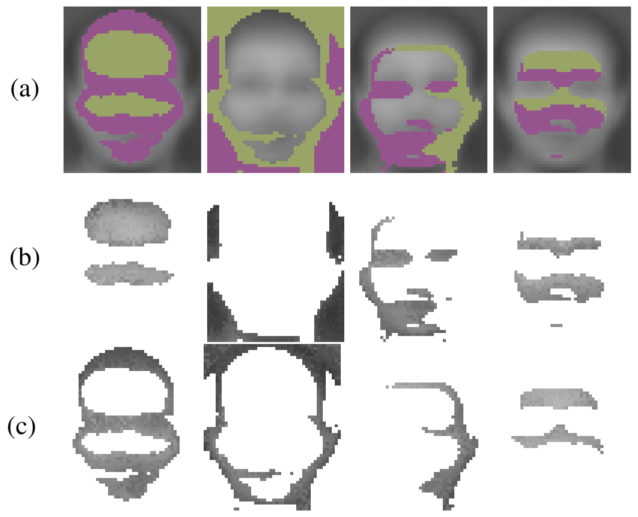

**Fig. 7**. D-SPNs on the Olivetti Faces dataset. Key facial features, such as eyes, nose, mouth and forehead are correctly identified and segmented.

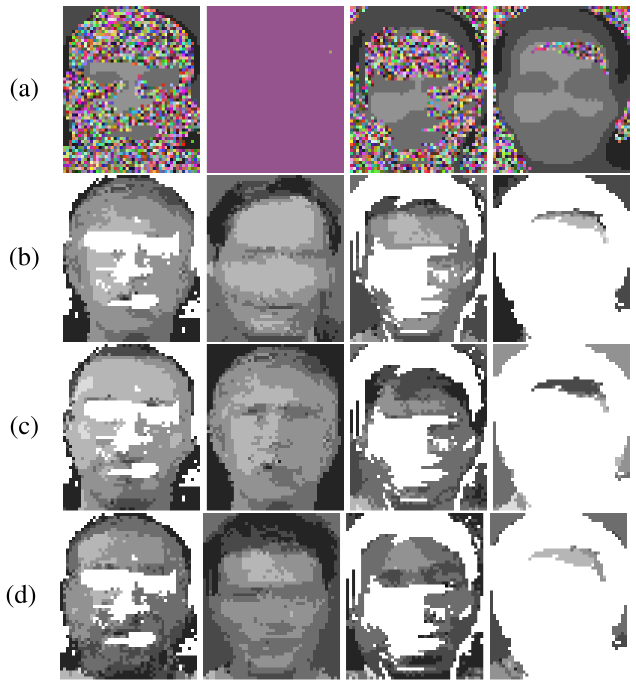

**Fig. 8**. Applying variation 2 of VisualizeSPN on G-SPNs learned with the Olivetti Faces dataset.

Applying VisualizeSPN to the Olivetti dataset yielded some interesting results. We found that SPNs were very capable of splitting facial features into significant regions. Figure 8 shows how the SPNs were able to learn how to segment the images to identify key facial regions. It was also able to distinguish between background and face, as evidenced by the second column. The third column suggests that the SPN was able to discriminate between left and right side. This is important, as the Olivetti dataset contains images with slightly turned faces. Reconstruction must take into account the face's side to provide accurate predictions. In G-SPNs case, it is not clear how the SPN identifies features, but it seems to provide human-like reconstructions as Figure 8 shows. It also seems to distinguish skin color better than D-SPNs.

## 5. CONCLUSION

We proposed a new technique of visualizing SPNs in the domain of image reconstruction. The method was applied on the Caltech-101 [13] and Olivetti Faces [14] dataset, on SPNs learned with both LearnSPN [2] and the Dennis-Ventura clustering algorithm [3]. We found this kind of visualization provided some interesting insights on how SPNs learn data.

## 6. REFERENCES

[1] Hoifung Poon and Pedro Domingos, “Sum-product networks: A new deep architecture,” *Uncertainty in Artificial Intelligence*, vol. 27, 2011.

[2] Robert Gens and Pedro Domingos, “Learning the structure of sum-product networks,” *International Conference on Machine Learning*, vol. 30, 2013.

[3] Aaron Dennis and Dan Ventura, “Learning the architecture of sum-product networks using clustering on variables,” *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[4] Robert Gens and Pedro Domingos, “Discriminative learning of sum-product networks,” *Advances in Neural Information Processing Systems*, pp. 3239–3247, 2012.

[5] Zehuan Yuan, Hao Wang, Limin Wang, Tong Lu, Shivakumara Palaiahnakote, and Chew Lim Tan, “Modeling spatial layout for scene image understanding via a novel multiscale sum-product network,” *Expert Systems with Applications*, vol. 63, pp. 231 – 240, 2016.

[6] M. R. Amer and S. Todorovic, “Sum-product networks for modeling activities with stochastic structure,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 1314–1321.

[7] M. R. Amer and S. Todorovic, “Sum product networks for activity recognition,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, April 2016, vol. 38, pp. 800–813.

[8] J. Wang and G. Wang, “Hierarchical spatial sum–product networks for action recognition in still images,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 1, pp. 90–100, Jan 2018.

[9] Antonio Vergari, Nicola Di Mauro, and Esposito Floriana, “Visualizing and understanding sum-product networks,” *Machine Learning*, 08 2016.

[10] Robert Peharz, Sebastian Tschiatschek, Franz Pernkopf, and Pedro M. Domingos, “On theoretical properties of sum-product networks,” in *AISTATS*, 2015.

[11] Jun Mei, Yong Jiang, and Kewei Tu, “Maximum a posteriori inference in sum-product networks,” in *AAAI*, 2018.

[12] Diarmaid Conaty, Denis D. Maua, and Casio P. de Campos, “Approximation complexity of maximum a posteriori inference in sum-product networks,” in *Proceedings of The 33rd Conference on Uncertainty in Artificial Intelligence*. 8 2017, AUAI.

[13] L. Fei-Fei, R. Fergus, and P. Perona, “Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories,” in *CVPR 2004, Workshop on Generative-Model Based Vision*. 2004, IEEE.

[14] Ferdinando Samaria and Andy Harter, “Parameterisation of a stochastic model for human face identification,” in *Proceedings of 2nd IEEE Workshop on Applications of Computer Vision*. 1994, IEEE.