Sum-Product Network and Its Application

Yilin ZHENG
11510506@mail.sustech.edu.cn
Southern University of Science and Technology

April 4, 2019

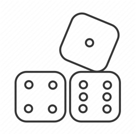
Outline

Outline

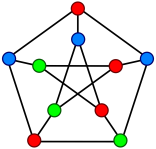
Background

Probabilistic graphical models(PGMs):

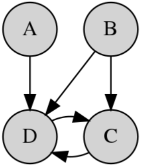
- Probability Theory
- Graph Theory



$+$



$=$



Classical PGMs

- Bayesian Network(BN): $p_{\mathcal{B}}(X) = \prod_{X \in \mathbf{X}} p(X|\mathbf{par}(X))$
- Markov Random Field(MRF):
  $p_{\mathcal{M}}(\mathbf{X}) = \frac{1}{\mathcal{Z}_{\mathcal{M}}} \prod_{l=1}^{L} \Psi_{\mathbf{C}_l}(\mathbf{X}_{\mathbf{C}_l})$

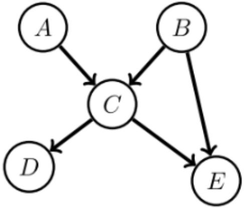


Figure: Example of a BN

$$p_{\mathcal{B}}(A, B, C, D, E) = p(A)p(B)p(C|A, B)p(D|C)p(E|B, C)$$

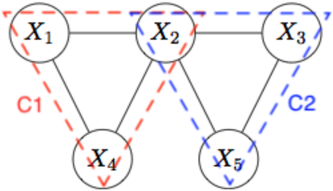


Figure: Example of a MRF

$$p_{\mathcal{M}}(X_1, X_2, X_3, X_4, X_5) = \frac{1}{\mathcal{Z}} \Psi_{\mathbf{C_1}}(X_1, X_2, X_4)\Psi_{\mathbf{C_2}}(X_2, X_3, X_5)$$

- Representation: joint distribution probability
- Learning: structure and parameters
- Inference: computing posterior marginal distributions

Classical PGMs suffer from some problems:

1. Scales unproportionally to the complexity of the model
2. Approximate learning yields unpredictable results
3. Intractable after slight modification
4. Separation in learning and inference

Sum-Product Network: Poon and Domingos(2011)

1. Scales up linearly
2. Exact learning
3. Tractability
4. Combination of learning and inference

Target

- Reproduce the SPN application on image completion
- Optimization of program
- Analysis the results

Roadmap

1. Literature review
2. Implementation of an SPN
3. Reproducing the experiment of image completion
4. Optimization and Analysis
5. Thesis draft and Defence preparation

Outline

Definition (Sum-Product Network)

- $\mathcal{S} = (\mathcal{G}, \Phi(\mathbf{X}))$, $\mathcal{G}$ is a rooted DAG and $\Phi(\mathbf{X})$ is a set of nonnegative parameters over random variables $\mathbf{X}$
- Leaves are numeric input, called *indicator variables*(IVs)
- Types of internal nodes: **sum** nodes or **product** nodes
- **Root** is a sum node
- Probability is computed at root
- Network polynomials of SPN: $f_{\mathcal{S}}(\mathbf{X}) = \sum_{X \in \mathbf{X}} \Phi(X) \prod_{X \in \mathbf{X}} (X)$

Example of an SPN

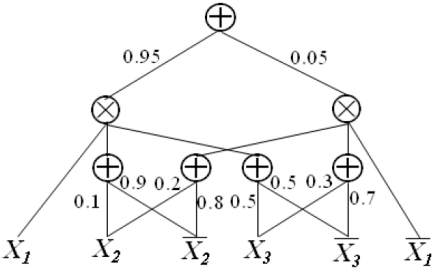


Figure: Example of an SPN with 6 indicator variables($\overline{X}$ represents negation of X)

$$
\begin{aligned}
f_{\mathcal{S}}(X_1, X_2, X_3, \overline{X}_1, \overline{X}_2, \overline{X}_3) = & 0.95X_1(0.1X_2 + 0.9\overline{X}_2)(0.5X_3 + 0.5\overline{X}_3) \\
& + 0.05(0.2X_2 + 0.8\overline{X}_2)(0.3X_3 + 0.7\overline{X}_3)\overline{X}_1
\end{aligned}
$$

Properties

Definition (Scope of a Node($\mathbf{sc}$))

$$\mathbf{sc}(N) \begin{cases} \{X\} & \text{if } N \text{ is a leaf} \\ \cup_{\mathbf{C} \in \mathbf{chi}(N)} \mathbf{sc}(\mathbf{C}) & \text{if } N \text{ is an internal node} \end{cases}$$

Properties:

- Completeness: a sum node's children have the same scope
- Consistency: a variable and its negation are in the same scope of a product node
- Validity: completeness + consistency
- Decomposability: no variable appears in more than one child of a product node node

Types of learning:

- Structure learning:
  - Poon-Domingos Architecture: Rectangle regions
  - Dennis-Venture Architecture: Any shape regions
- Parameter learning:
  - Gradient method: maximize log-likelihood
  - EM algorithm: introduce latent variables, inference in E-step, update weights in M-step

Differential approach:

1. Compute the marginal distribution via network polynomials
2. Interpret the probability distribution through derivatives

Outline

Learn the half image and recover the left half image by learning and inference over the SPN constructed on given dataset.

Poon-Domingos Architecture

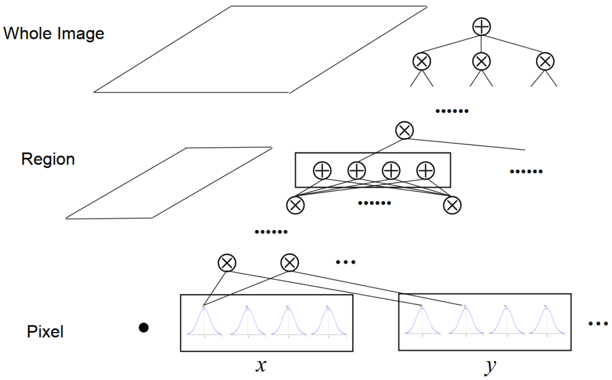


Figure: Poon-Domingos Architecture

Program

- **common**: Contain some helper functions to provide time management, messaging between progress(MPI), parameter settings for training on clusters, and some utilities.
- **evaluation**: Process the dataset, apply network to datatset to output models, and evaluate the results generated from the models.
- **spn**: Contain the SPN architecture, including node definition, computation functions, the learning, and inference.

Files: 16 .cpp, total code: 3605 lines

Callgraph

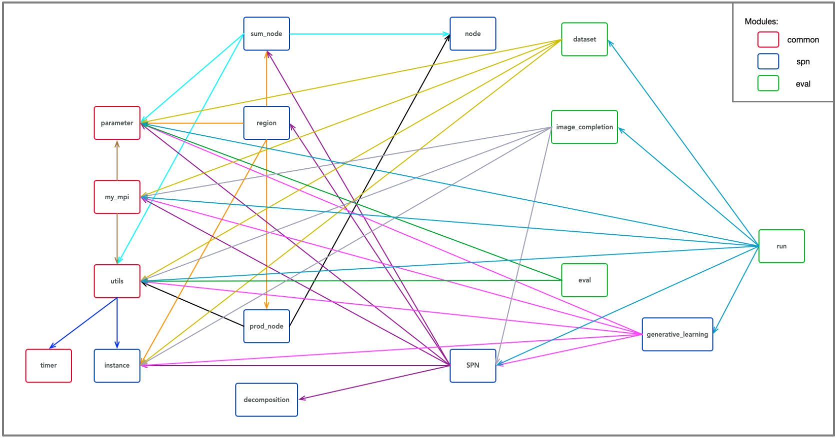


Figure: Callgraph of Experiment Program

Platform

Platform: TaiYi cluster
Cores:

- Caltech: 120
- Olivetti: 80

Caltech:

- 101 categories, from 40 to 800 images per category
- about 300×200 pixels, rescaled to 100×64 pixels in the experiment.

Olivetti:

- face images taken between April 1992 and April 1994 at AT&T Laboratories Cambridge
- image size 64×64 pixels

Outline

Done

- A brief review of the SPN
- Monthly report till Mar. 2019
- Implementation of SPN architecture
- Unit Tests of the Program
- Interim Report

Processing

- Debug of learning part of the program
- Reproduce the results of image completions
- Draft the thesis

ToDo

- Optimization of codes
- Compare and analysis the results

Thank you for your listening! ☺