Sum-Product Network and Its Application to Image Completion

A thesis defense

Yilin Zheng
Supervised by: Ke Tang & Shan He

Southern University of Science and Technology(SUSTech)

May 20, 2019

Contents

Traditional PGM: Bayesian Network

Bayesian Network(BN):

- Graph type: directed acyclic graph(DAG)
- Representation: conditional dependence
- Network polynomial: $p_{\mathcal{B}}(X) = \prod_{X \in \mathbf{X}} p(X|\mathbf{parent}(X))$

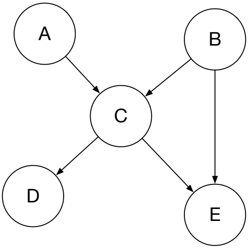


Figure: An Example of BN

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

Traditional PGM: Markov Random Field

Markov Random Field(MRF):

- Graph type: undirected graph
- Representation: Markov properties
- Network polynomial: $p_{\mathcal{M}}(\mathbf{X}) = \frac{1}{\mathcal{Z}_{\mathcal{M}}} \prod_{l=1}^{L} \Psi_{\mathbf{C}_l}(\mathbf{X}_{\mathbf{C}_l})$

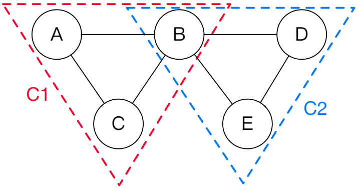


Figure: An Example of MRF

$$p_{\mathcal{M}}(A, B, C, D, E) = \frac{1}{\mathcal{Z}} \Psi_{\mathbf{C}_1}(A, B, C) \Psi_{\mathbf{C}_2}(B, D, E)$$

Learning

Learning:

1. Object:
   - Structure learning
   - Parameters learning
2. Approach:
   - Generative learning
   - Discriminative learning

Inference

Inference:

- Marginalization
- Conditions
- Most probable explanation(MPE)
- Maximum a-posterior(MAP)

Weaknesses

Weaknesses:

1. Complexity scales unproportionally
2. Approximate learning
3. Intractability
4. Separation of learning and inference

Motivation

Why SPN:

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

Target

Targets:

1. Implement an SPN
2. Reproduce the application to image completion

What is SPN

Sum-product network(SPN):

- Graph type: DAG
- Leaves: random variables
- Internal node: sum node, product node
- Root: sum node
- Network polynomial: $f_{\mathcal{S}}(\mathbf{X}) = \sum_{X \in \mathbf{X}} \Phi(X) \prod_{X \in \mathbf{X}} (X)$

Example

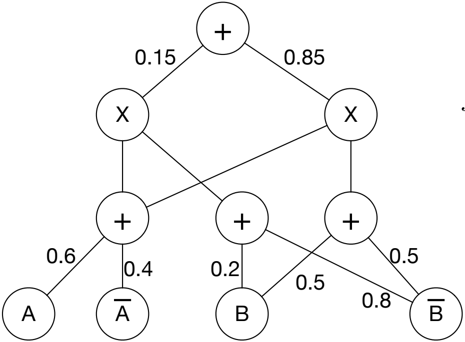


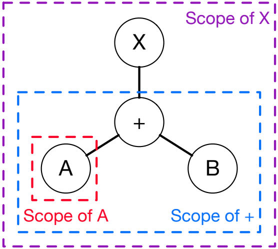
Figure: An Example of SPN

$$
\begin{aligned}
&f_S(A, B, \overline{A}, \overline{B}) \\
=&0.15(0.6A + 0.4\overline{A})(0.2B + 0.8\overline{B}) \\
+&0.85(0.6A + 0.4\overline{A})(0.5B + 0.5\overline{B}) \\
=&0.273AB + 0.327A\overline{B} + 0.182\overline{A}B \\
+&0.218\overline{AB}
\end{aligned}
$$

Scope

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

Properties:

- Completeness
- Consistency
- Validity
- Decomposability

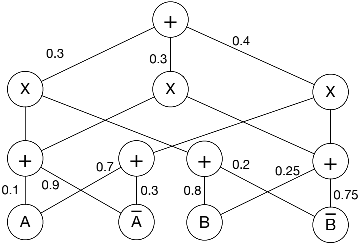


Figure: An Example of Valid SPN

Learning Methods

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

Inference Methods

Differential approach:

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

Poon-Domingos Architecture

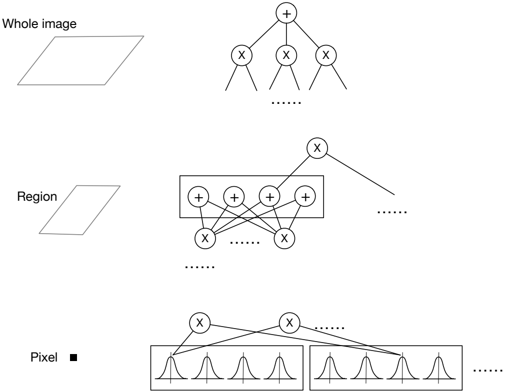


Figure: Poon-Domingos Architecture

Program

Code:

- **common**: helper functions for time management, messaging between progress(MPI), parameter settings, and some utilities.
- **evaluation**: dataset processing, model generation, evaluation.
- **spn**: SPN architecture, including node definition, computation functions, the learning, and inference.

Enviroment

Enviroment:

- Platform: TaiYi
- Library: OpenMPI C++
- Dataset:
  - **Caltech**:
    - 101 categories, from 40 to 800 images per category
    - about 300×200 pixels, rescaled to 100×64 pixels
  - **Olivetti**:
    - face images taken between Apr. 1992 and Apr. 1994 at AT&T Laboratories Cambridge
    - size 64×64 pixels

Experiments

Experiments:

1. **Caltech**: 80 cores, **Olivetti**: 40 cores, size 64×64
2. **Caltech**: 120 cores, **Olivetti**: 80 cores, size 64×64
3. **Caltech**: 80 cores, size 100×64
4. **Caltech**: 120 cores, size 100×64

Poon's experiments: 102 cores, **Olivetti**: 51 cores, size 64×64

Comparison on MSE(1)

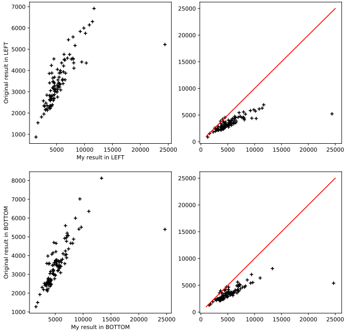


Figure: Exp. #1 vs Poon's(Caltech)

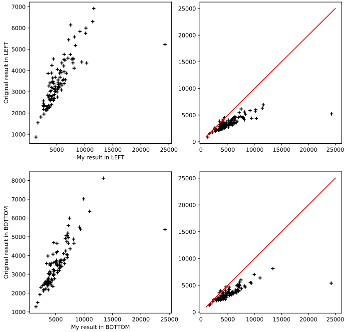


Figure: Exp. #2 vs Poon's(Caltech)

Comparison on MSE(2)

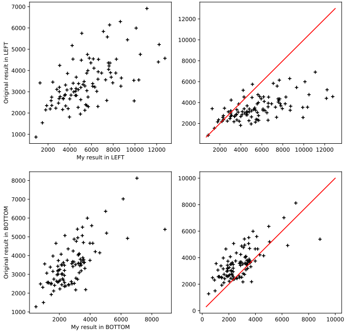


Figure: Exp. #3 vs Poon's(Caltech)

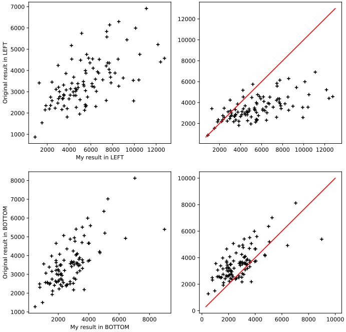


Figure: Exp. #4 vs Poon's(Caltech)

Comparison on Number of Cores

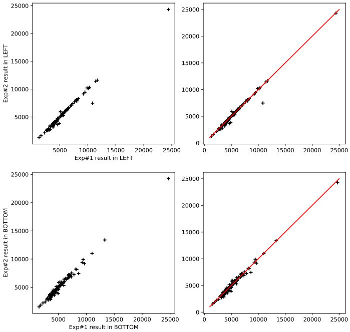


Figure: Exp. #1 vs Exp. #2(Caltech)

Comparison on Input Size

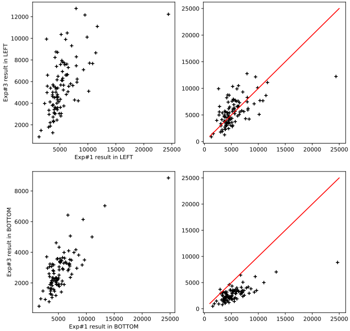


Figure: Exp. #1 vs Exp. #3(Caltech)

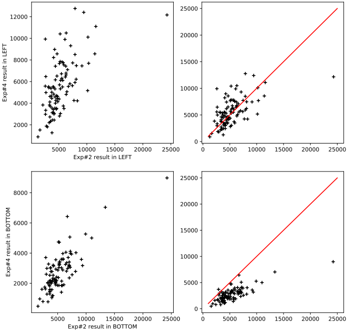


Figure: Exp. #2 vs Exp. #4(Caltech)

Comparison on Image(1)

Figure: Airplanes-bottom(Poon's, Exp. #2, Exp. #4)

Figure: Yin_yang-left(Poon's, Exp. #2, Exp. #4)

Comparison on Image(2)

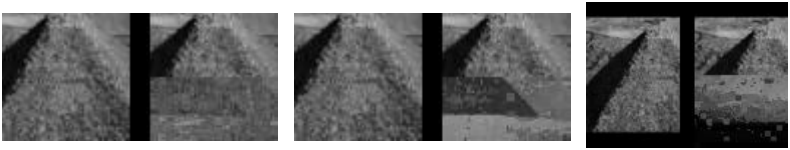


Figure: Pyramid-bottom(Poon's, Exp. #2, Exp. #4)

Figure: Sunflower-bottom(Poon's, Exp. #2, Exp. #4)

Comparison on Time

Time Cost	Poon's	Exp. #1	Exp. #2	Exp. #3	Exp. #4
Caltech-101	≤ 2 hours	≤ 4 hours	≤ 7 hours	≤ 11 hours	≤ 19 hours
Olivetti	within a few minutes	≤ 4 minutes	≥ 72 hours	Nan	Nan

Table: Time Cost of Poon's Experiments and My Experiments

Analysis

Conclusion:

- Number of cores makes no influence
- Larger input size leads to lower MSE
- My implementation is valid

Why:

- Randomness in architecture
- Difference between implementation
- Complexity of model

Conclusion

Conclusion:

- My implementation is valid and successful
- Reproduction is not easy

Future Work

Future work:

1. Architecture improvement
2. More applications
3. New algorithms for learning and inference

Acknowledgement

Advisors:

Prof. Ke Tang
Prof. Shan He

Inspector:

Prof. Bo Tang

Committee Members:

Prof. Qi Wang (Committee chair)
Prof. Jianqiao Yu (Committee member)
Prof. Jialin Liu (Committee member)

Thanks

Thanks for listening!

Q & A

Questions ?