A Brief Report of My Thesis Progress

Yilin ZHENG

11510506@mail.sustc.edu.cn

Southern University of Science and Technology

January 29, 2019

Contents

Chapter 1

Project Description

1.1 About this Project

This project, as my undergraduate thesis, is to implement an efficient SPN and conduct research on a Gene Dataset if possible. This project supervised under Prof. He and Prof. Tang, with the assistant of Mr. Lee, a Doctoral candidate supervised by Prof. He.

1.1.1 Roadmap

Roadmap of this project can be briefly described as three stages:

1. Reproduce the result of original paper which firstly proposed SPN.

2. Optimize the codes and move the architecture to apply to the specified dataset.

3. Draft thesis and prepare the defence

Note that Stage 1 is to validate the implementation of SPN architecture over C++ and the results derived from my implementation, and Stage 2 might be skipped if time

and resource are unavailable.

1.1.2 Target

Two targets are arranged in sequence:

1. Reproduce the result presented in the original paper, which is the application of SPN on image completion.

2. Optimize the code and move to apply to our specified dataset.

1.2 Current Progress

Currently, an architecture of SPN in C++ has been implemented. According to the discussion with Prof. He and Mr. Lee, the implementation need to be validated over original dataset which might be more helpful to my thesis.

Chapter 2

Codes

2.1 Code Structure

The source code of the SPN architecture is divided into three folders:

- **common**: Contain some helper functions to provide time management, messaging between progress, parameter settings, and some utilities.
- **eval**: Process the dataset, apply network to datatset, and evaluate the result.
- **spn**: Contain the modules of SPN, including node definition, computation functions, and the learning process.

2.2 Documents

Code docs are described in modules respectively:

- **common**

1. *my_mpi*: Use **OpenMPI** to support the messaging in a parallel program. It means that this program will use parallel architecture to accelerate computing.
2. *parameter*: Control parameters for EM algorithm, SPN, and evaluation.
3. *timer*: Manage the time to help calculate the time spent on computation.
4. *utils*: Some helper functions to access time, print log, and do some numeric process.

- **eval**

1. *dataset*: Read and process data from the dataset.
2. *eval*: Conduct evaluation over the dataset.
3. *image_completion*: Conduct image completion, which is the application.
4. *run*: Control the program, which is the main function.

- **spn**

1. *decomposition*: Decompose the regions.
2. *generative_learning*: Conduct the learning process to generative the model.
3. *instance*: Record the mean and variance of an instance, which is calculated from the dataset.
4. *node*: Define the node, to provide the base class of the sum node and the product node.
5. *prod_node*: Define the product node, derived from *node*.
6. *sum_node*: Define the sum node, derived from *node*.

7. *region*: Compute the mean and variance of regions in the picture(for this image completion application), as well as the MAP.

8. *SPN*: Define the Sum-product network, including a root, functions of learning and applications. These functions are implemented via calling other modules.

2.3 Callgraph

The callgraph is presented in Figure 2.1

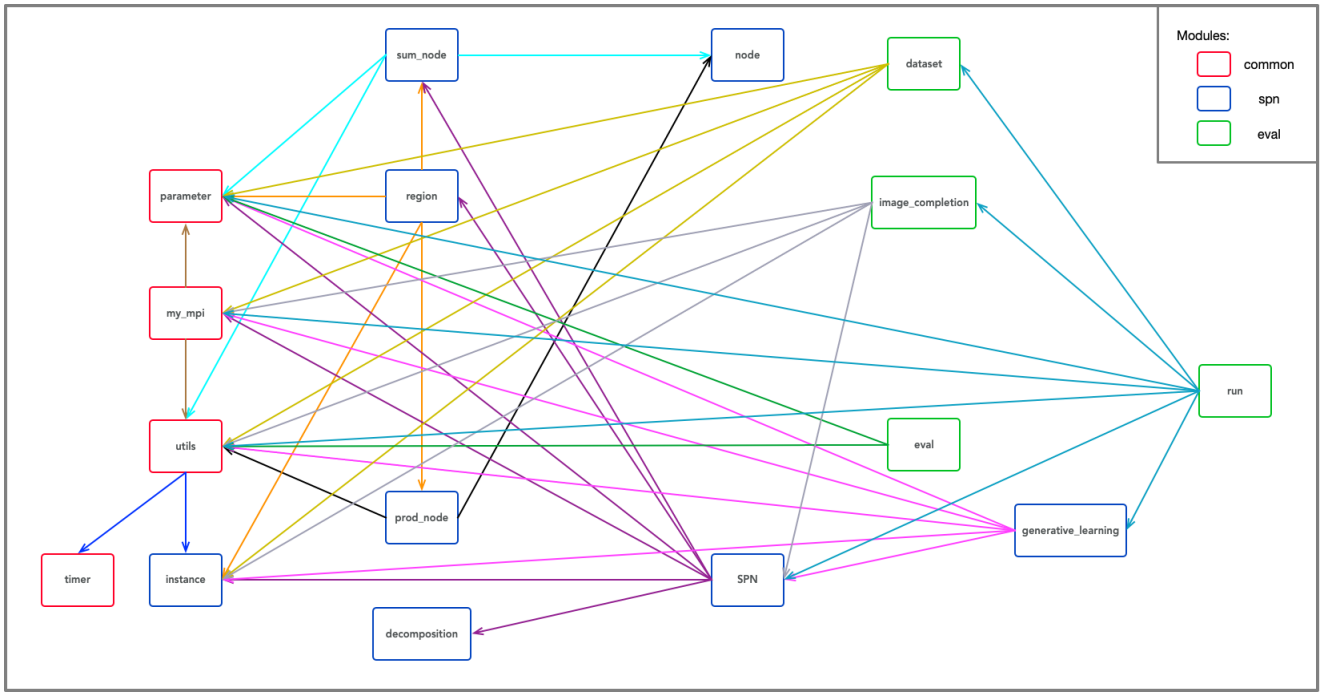

Figure 2.1: Callgraph of SPN

Chapter 3

Future Work

3.1 What to DO

Since the original result has not been reproduced, I will Since the original result has not been reproduced, I will continue to finish reproducing the original application. That means the modules of evaluation will be implemented next step and I will try to make the program runnable on servers. Stage 2 will be considered according to my future progress.