Replicating Process Discovery with Graph Neural Networks with Real Datasets

Yuhan Situ, 614651
Bin Zeng, 621286
Kunpeng Xiao, 638214

February 5, 2024

Abstract

In a research paper from a university in the Netherlands, researchers endeavor to explore challenges in supervised learning of process discovery techniques. [1] They propose a technique utilizing graph convolutional neural networks to train a machine learning-based model. This model translates given input event logs into a sound Petri net. The objective is to train the model on synthetically generated pairs of input logs and output models, enabling it to effectively transform previously unseen synthetic and various real-life event logs into models with accuracy and simplicity comparable to existing state-of-the-art techniques. Our task is to replicate the most crucial part of this methodology. This report documents our experimental procedure and results, and the final section contains a comparison and discussion between us and the original study.

1 Introduction

Graph Neural Networks (GNNs) represent a category of machine learning models designed specifically for handling graph data. They focus on learning relationships between nodes and edges in a graph, addressing tasks such as node classification and graph classification. Through iterative information propagation and aggregation, GNNs allow nodes to progressively consider local and global contexts, proving effective for tasks involving complex network structures, such as social networks and recommendation systems.

2 Task

Our task is to replicate the most crucial part of this methodology.

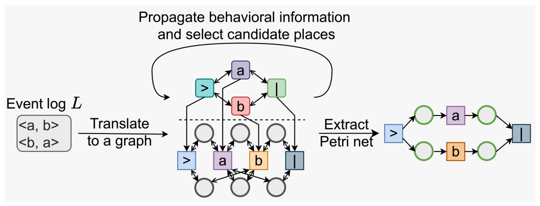

Figure 1: high-level overview of the entire model

3 Approaching

3.1 Experiment preparation

All experiments are performed on an RTX 4090 GPU (24GB) with 32 GB RAM. And the source is available at gitlab.com/dominiquesommers/apd-ml.

3.2 Experimental pipeline

First, the model transforms the event log L into a graph G containing the event log L and candidate locations P of the candidate Petri nets N. The model's learning task is then to generate the final model N by selecting the candidate locations P that are most likely to explain the behavioral information in the event log L. The model generates the final model N by iteratively propagating the behavioral information in the event log into the model structure of the candidate Petri nets N, and selecting candidate locations one by one to learn how to solve this discovery problem. A high-level overview of the entire model is shown in Figure 1, which details the workflow of the model. .

1. Train Model

   Training arguments with: Embedding size = 21, Embedding strategy=one hot, Train/Test set split ratio = 0.75, epoch = 500, learning ratio = 0.0005, etc..

2. Generate Petri-Nets

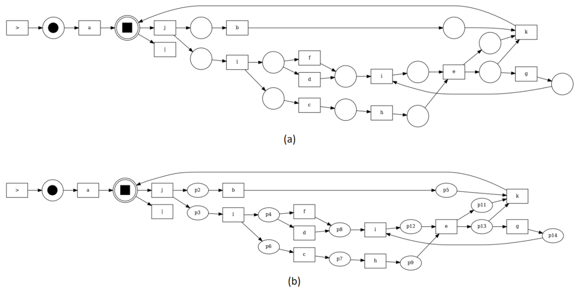
Figure 2: Example of Generated Petri-Nets

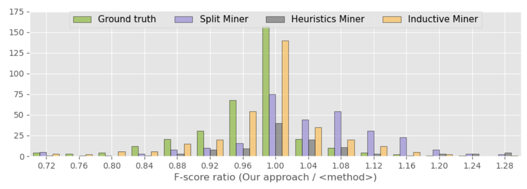

Figure 3: Ratio of F-score

Use the trained GNN model to generate Petri-Net and compare to the ground truth in Figure 2. . Where fig.(a) is the petri-net generated by the model and fig.(b) is the ground truth respectively. It is rather clear to see that the model already has good capability to generate proper petri-net from the log.

3. Obtain Results

   To show the capability of the model, several methods have been conducted and will be shown in the following section.

4 Results

The paper employed F-scores to assess the performance of a specific method. The detailed definition of the F-score has been omitted due to the page limitations.

Figure 3 illustrates the F-score ratio between the GNN method and alternative methods on the road traffic line dataset. The superiority of the GNN model over other models is evident.

Figure 4 provides insights into GNN models from multiple dimensions, incorporating seven different methods. The comparison of histograms reveals that the GNN model (depicted in blue) attains elevated simplicity scores, albeit with a slight compromise in accuracy. This suggests a more concise nature of the model.

Additionally, another figure can be generated to draw a conciser understanding of the model's performance. Figure 5 illustrates medians and means for each method, comparing them with the ground

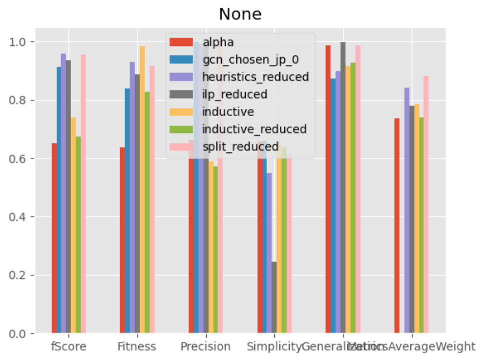

Figure 4: Histogram for Different Perspectives

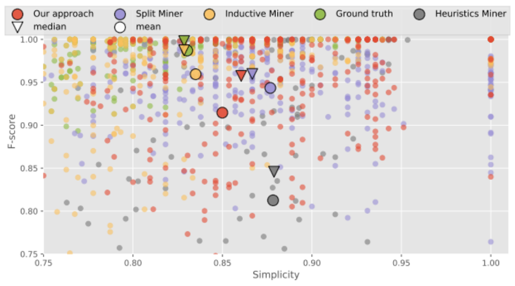

Figure 5: Simplicity

truth. The discerned gap between the median and mean in the GNN approach implies varied conformance levels across samples, offering valuable insights into its performance.

5 Discussion

Through training on synthetic data, the GNN model shows competence in handling new synthetic and real-life logs, generating models of comparable accuracy and simplicity to existing methods. The study reflects on the trade-off between accuracy and simplicity, offering insights into prioritizing simplicity without sacrificing substantial accuracy.

Drawing upon prior knowledge in process mining, it is evident that logs typically consist of a limited set of events, rendering them less intricate. This prompts consideration of whether there exists variants of GNN models for generating simpler graph structures. Exploring the potential of simpler GNNs in future research may yield enhanced simplicity in process mining, presenting an intriguing avenue for future investigation.

References

[1] Dominique Sommers, Vlado Menkovski, and Dirk Fahland. Process discovery using graph neural networks. *CoRR*, abs/2109.05835, 2021.