

The Effects of Node Removal on Bayesian Network Resilience for ATM Network Transaction Vulnerabilities

Gcobisile Matafeni
709637

UNIVERSITY OF THE
WITWATERSRAND,
JOHANNESBURG



A dissertation submitted to the Faculty of Science,
University of the Witwatersrand, in fulfillment of the requirements
for the degree of Master of Science.

Supervised by Prof. Ritesh Ajoodha and Dr. Seun Olukanmi

May 30, 2025

Abstract

We investigate the evaluation of influence relationships in probabilistic graphical models, focusing on the impact of node removal (mutilation) within Bayesian networks. The central problem addressed is understanding how the joint probability distribution and influence structure among interconnected variables evolve when a subset of nodes is removed, an issue relevant to various real-world systems experiencing disruptions. We model these dynamics using Bayesian learning to provide insights into network resilience and dependencies. To explore these effects, we generate synthetic Bayesian network structures that are tree-like, sparse, and dense, each representing different real-world configurations found in machine learning, sensor networks, and financial modeling. Conditional Probability Distributions (CPDs) were assigned to nodes based on the Bernoulli distribution. The Kullback-Leibler (KL) divergence quantified the deviations in influence structures post-removal, with evaluation of structure recovery employing an exact inference technique. Our findings indicate that each network type exhibits distinct responses to node removal: tree-like structures stabilize quickly with increased data, sparse structures show higher sensitivity but recover efficiently, and dense structures offer robustness through redundancy, though they demand larger datasets. These findings have significant implication for optimizing complex systems, particularly those requiring resilient network architectures. As a real-world application, we model ATM transaction networks to analyze how the removal of ATMs (due to vandalism, load shedding, or maintenance) impacts transaction flows. Our results show that high-traffic ATMs serve as critical nodes, significantly influencing neighboring ATMs when removed. By applying Bayesian structure learning, we demonstrate that optimal ATM network configurations can be identified to minimize disruption and improve financial service resilience. This study contributes to the growing field of probabilistic graphical models by introducing a novel approach to understanding influence dynamics in mutilated networks. It provides practical insights and lays a foundation for further research into complex systems where node integrity and network stability are critical for decision-making and operational efficiency.

Keywords: Bayesian network, probabilistic graphical models, mutilation, influence relationships, node removal, ATM network resilience, Kullback-Leibler divergence, structure learning

Declaration

I, Gcobisile Matafeni, hereby declare the contents of this masters dissertation to be my own work. This dissertation is submitted for the degree of Master of Science in Computer Science at the University of the Witwatersrand. This work has not been submitted to any other university, or for any other degree.

Signature: *G. Matafeni*
Gcobisile Matafeni

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Prof. Ritesh Ajoodha, and my co-supervisor, Dr. Seun Olukanmi, for their unwavering support, invaluable guidance, and continuous motivation throughout my research journey. Their expert insights and constructive feedback have been instrumental in shaping the direction and quality of this dissertation.

I am also deeply grateful to Rudzani Mulaudzi from the Explainable AI Lab, whose assistance with my coding challenges provided crucial breakthroughs when I felt stuck. His willingness to share his expertise significantly enhanced my understanding and implementation of the experimental framework.

My heartfelt thanks go to Ankan, Ankur, Abinash, and Panda for their efforts in developing the Python pgmpy framework, which served as the foundation for the experimental implementation in this study. Their work not only facilitated my research but also enriched my practical experience with advanced probabilistic graphical models.

Finally, I wish to acknowledge my family and friends, whose continuous encouragement, understanding, and support have been a constant source of strength. I am especially thankful to my family, to whom I dedicate this dissertation, for believing in my potential and for always being there for me.

Contents

Abstract	i
Declaration	ii
Acknowledgements	iii
Table of Contents	iv
Contents	iv
List of Figures	vi
List of Tables	1
1 Introduction	2
2 Background and Related Work	4
2.1 Introduction	4
2.2 Reasoning Patterns in Bayesian Modeling	4
2.3 Forecasting Techniques for ATM Networks	6
2.4 Optimizing Cash Flow and Resource Allocation	11
2.5 Related Work	14
2.5.1 Mapping Influence in Complex Systems	14
2.5.2 Conclusion	16
3 Methodology	17
3.1 Problem Formulation	17
3.2 Research Questions and Experimental Setup	18
3.2.1 Experimental Setup	18
3.3 Application Domain	19
3.3.1 Modeling Influence Relationships	19
3.4 Mutilating the Network	20
3.4.1 Learning	21
3.5 Evaluation	23
3.6 Modeling the Influence of Transaction Volumes between ATMs	26
3.6.1 Data Preprocessing	27
3.6.2 Data and Ethical Implications	27
3.6.3 Practical Implementation and Challenges	28
3.7 Conclusion	29
4 Experimental Results and Discussion	30

4.1	Research Questions 1 and 2: What is the impact on the KL divergence when nodes are removed from a recovered Bayesian network? & What are the Implications of Varying Structural Complexity on the Recovery of Bayesian Networks and their KL Divergence?	30
4.1.1	Tree-like Structures	31
4.1.2	Sparse Structures	37
4.1.3	Dense Structures	45
4.1.4	Comparative Analysis and Synthesis Across Network Models	51
4.2	Research Question 3: How can Insights from Bayesian Networks Improve the Operational Efficiency and Planning of ATM Networks?	52
4.2.1	Comparing Original and Reduced Networks	61
4.3	Conclusion	62
5	Conclusion and Future Work	64
	References	70

List of Figures

1.1	A bombed/vandalised Automated Teller Machine (The Citizen, 2017).	2
2.1	Different influence structures that depict the influence relationships of node Z with other nodes.	4
2.2	A Bayesian network influence structure	5
2.3	A Bayesian network influence structure with a removed node (Z).	5
3.1	An influence structure between stochastic processes. The red arrows indicate the influence between these processes while the solid arrows indicate the parameters of each node.	20
3.2	A visual representation of a dataset with θ representing the parameters.	22
3.3	An influence structure between stochastic processes with a reduced graph structure. . . .	24
3.4	A distribution of ATM networks in the Johannesburg region for a South African bank. The yellow circles with white stars in the center indicate an individual ATM, the red crosses on these ATMs indicate a hypothetical event of removing an ATM from a network.	27
4.1	10 randomly generated tree-like DAGs with random edge orientations and random Bernoulli CPDs assigned to each node.	31
4.2	A snapshot of the top 10 samples that are generated from one of the GTD DAGs.	32
4.3	A plot of the training data versus the KL divergence for GTDs with 150 nodes comparing the Tree Search and MWST algorithms.	32
4.4	A plot of the training data versus the KL divergence for GTDs with 10 nodes using MWST for structure learning.	33
4.5	A plot of the training data versus the KL divergence for GTDs with 10 nodes using MWST for structure learning with mutilation.	33
4.6	A plot of the training data versus the KL divergence for GTDs with 50 nodes using MWST for structure learning.	34
4.7	A plot of the training data versus the KL divergence for GTDs with 50 nodes using MWST for structure learning with mutilation.	34
4.8	A plot of the training data versus the KL divergence for GTDs with 100 nodes using MWST for structure learning.	35
4.9	A plot of the training data versus the KL divergence for GTDs with 100 nodes using MWST for structure learning with mutilation.	35
4.10	A plot of the training data versus the KL divergence for GTDs with 150 nodes using MWST for structure learning.	36
4.11	A plot of the training data versus the KL divergence for GTDs with 10 nodes using MWST for structure learning with mutilation.	36
4.12	10 randomly generated sparse DAGs with random edge orientations and random Bernoulli CPDs assigned to each node.	38

4.13	A snapshot of the top 10 samples that are generated from one of the GTD DAGs.	38
4.14	A plot of the training data versus the KL divergence for GTDs with 10 nodes comparing the BIC, KIC and BDScore for scoring while using the Hill Climb Search algorithm for structure learning.	39
4.15	A plot of the training data versus the KL divergence for GTDs with 10 nodes using Hill Climb Search with BIC for scoring.	40
4.16	A plot of the training data versus the KL divergence for GTDs with 10 nodes using Hill Climb Search with BIC for structure learning with mutilation.	40
4.17	A plot of the training data versus the KL divergence for GTDs with 15 nodes using Hill Climb Search with BIC for scoring.	41
4.18	A plot of the training data versus the KL divergence for GTDs with 15 nodes using Hill Climb Search with BIC for structure learning with mutilation.	42
4.19	A plot of the training data versus the KL divergence for GTDs with 20 nodes using Hill Climb Search with BIC for scoring.	42
4.20	A plot of the training data versus the KL divergence for GTDs with 20 nodes using Hill Climb Search with BIC for structure learning with mutilation.	43
4.21	A plot of the training data versus the KL divergence for GTDs with 25 nodes using Hill Climb Search with BIC for scoring.	43
4.22	A plot of the training data versus the KL divergence for GTDs with 25 nodes using Hill Climb Search with BIC for structure learning with mutilation.	44
4.23	10 randomly generated dense DAGs with random edge orientations and random Bernoulli CPDs assigned to each node.	45
4.24	A plot of the training data versus the KL divergence for GTDs with 10 nodes comparing the BIC, KIC and BDScore for scoring while using the Hill Climb Search algorithm for structure learning.	46
4.25	A plot of the training data versus the KL divergence for GTDs with 10 nodes using Hill Climb Search with BIC for scoring.	46
4.26	A plot of the training data versus the KL divergence for GTDs with 10 nodes using Hill Climb Search with BIC for structure learning with mutilation.	47
4.27	A plot of the training data versus the KL divergence for GTDs with 15 nodes using Hill Climb Search with BIC for scoring.	47
4.28	A plot of the training data versus the KL divergence for GTDs with 15 nodes using Hill Climb Search with BIC for structure learning with mutilation.	48
4.29	A plot of the training data versus the KL divergence for GTDs with 20 nodes using Hill Climb Search with BIC for scoring.	48
4.30	A plot of the training data versus the KL divergence for GTDs with 20 nodes using Hill Climb Search with BIC for structure learning with mutilation.	49
4.31	A plot of the training data versus the KL divergence for GTDs with 25 nodes using Hill Climb Search with BIC for scoring.	49
4.32	A plot of the training data versus the KL divergence for GTDs with 25 nodes using Hill Climb Search with BIC for structure learning with mutilation.	50
4.33	A plot of the total transaction volumes for 156 ATMs starting from January 2021 until November 2024.	52
4.34	A heatmap of transaction volumes for first 30 days for the first 10 ATMs.	53
4.35	A Box Plot of transaction volumes for the first 5 ATMs.	53
4.36	A plot of the distribution of transaction volumes for ATM 1.	54
4.37	A snapshot of the quantized ATM transaction data.	54
4.38	A recovered Bayesian Network for the first 20 ATMs.	55

4.39	A recovered Bayesian Network for the first 20 ATMs overlaid on a GIS map with their longitude and latitude.	55
4.40	Evaluation metrics applied on 3 ATMs.	56
4.41	Reduced graph structure with the removal of ATM 15	56
4.42	Reduced graph structure with the removal of ATM 15 overlaid on a GIS map.	57
4.43	Evaluation metrics on two ATMs after the graph is reduced.	57
4.44	A recovered Bayesian Network for the first 156 ATMs.	58
4.45	Snapshot of a recovered Bayesian Network for 156 ATMs.	58
4.46	Evaluation metrics applied on ATM 62, ATM 105 and ATM 3 and ATM 20.	59
4.47	Reduced graph structure with the removal of ATM 15.	59
4.48	Reduced graph structure with the removal of ATM 15.	60
4.49	Evaluation metrics applied on ATM 62, ATM 105 and ATM 3 and ATM 20.	60
5.1	An influence structure between stochastic processes with a reduced graph structure for a temporal model set up.	65

List of Tables

- 2.1 A table comparing different forecasting techniques that were used by other authors. . . . 10
- 2.2 A table comparing different optimization techniques that were used by other authors. . . . 13
- 2.3 A table comparing different influence modeling techniques that were used by other authors. 16

Chapter 1

Introduction

Numerous scientific phenomena can be conceptualized as stochastic processes, which characterize the evolution of random variables over time or space. Employed for modeling and analysing random occurrences, stochastic processes offer insights into the interactions of complex systems amidst uncertainty. By treating each process as a distinct stochastic process with variables, we can discern the relationships and influences between them. The relationships between these variables can be expressed with a joint probability distribution which models the influence between these variables. Given a set of these variables and a joint probability distribution that describes the influence between them, how would removing a variable or a subset of variables affect the influence structure and the new joint probability distribution of the remaining variables?

A grouping of Automated Teller Machines (ATMs) can be viewed through the lens of stochastic processes. ATMs, which are prevalent in society, serve as key tools for financial transactions. They offer customers the ability to withdraw and deposit money, providing convenience and various other benefits to both banks and customers. Due to their widespread deployment across geographically dispersed locations, ATMs present banks with logistical and financial optimization challenges. Analyzing the connections between ATMs can yield insights applicable to a range of issues encountered by financial institutions. If an ATM or a subset of ATMs in a network of ATMs were to go off due to ATM bombing, load shedding, planned maintenance etc, how would this affect the rest of the ATMs in the same region, would customers move to transact in the nearest ATM, what associations manifest in the data and how would we go about detecting vulnerabilities in the network?



Figure 1.1: A bombed/vandalised Automated Teller Machine (The Citizen, 2017).

Historically, previous studies examining influences between ATMs have primarily focused on forecasting transactions and optimizing various operational aspects, such as predicting daily cash demand, scheduling efficient cash replenishment, and strategically placing ATMs, in order to enhance overall network efficiency and customer service. Different techniques have been explored for predicting the daily replenishment of ATMs, namely regression approaches and Long Short-Term Memory. Simutus *et al.* [2007a] tried to solve the same problem by utilizing neural networks and support vector regression techniques which performed considerably well with forecasting cash demand. Genevois *et al.* [2015] looked at the ATM location problem concurrently with cash management by utilizing a cash demand model for every ATM in a network.

Ajoodha [2018] used dynamic Bayesian networks to model the influence between stochastic processes by making use of score-based structure learning. Ajoodha [2018] devised an algorithm that recovers the influence relations between a set of stochastic processes. Ajoodha and Rosman [2020] provided a method for learning the influence between processes that are represented by hidden Markov models (HMM) as opposed to dynamic Bayesian networks. In our study, we aim to explore how the influence among variables evolves when a variable or subset of variables is eliminated. This approach has not been previously explored in the existing literature that has been examined.

This study aims to explore the below questions by investigating the effect of node removal in Bayesian networks. The key research questions guiding this investigation are as follows:

1. What is the impact on the KL divergence when nodes are removed from a recovered Bayesian network? This question examines how the probabilistic structure and information content of a Bayesian network change when certain nodes and their connections are removed. By quantifying the deviation in learned distributions using KL divergence, we evaluate the network's ability to recover its original properties post-mutilation.
2. What are the implications of varying structural complexity on the recovery of Bayesian networks and their KL divergence? To answer this, we analyze different types of Bayesian networks starting with tree-like, sparse, and dense structures, to determine how structural differences influence resilience to node removal. This comparison helps identify which network types are more robust to disruptions.
3. How can insights from Bayesian networks improve the operational efficiency and planning of ATM networks? Applying Bayesian modeling to ATM transaction networks allows us to study how transaction flows change when ATMs go offline. By identifying critical nodes and their influence, we provide recommendations for optimizing ATM placement and mitigating service disruptions.

Our principal contribution is to show how the joint probability distribution and influence structure of a Bayesian network change when a subset of nodes is removed. To accomplish this, we begin by creating a random Bayesian network, where the nodes are connected to form a directed acyclic graph and each node is assigned a Conditional Probability Distribution using a Bernoulli distribution. We then generate random samples to form a dataset, learn the probability patterns, and systematically remove nodes to re-learn the altered patterns. This process allows us to quantify how network influence and structure are affected by node removal.

The remainder of the dissertation is structured as follows. Chapter 2 presents the background and related work to Bayesian learning. In the chapter, we dive into detail about the methods that have been used to solve similar problems. The research problem is stated in Chapter 3 as well as the methodology that is used to solve for it. Chapter 4 focuses on the experimental results and discusses the limitations of our work. Chapter 5 wraps up the whole research dissertation.

Chapter 2

Background and Related Work

2.1 Introduction

The previous chapter introduced the general problem area and the specific problem area for this study. In this chapter we review works relating to modeling the influence between stochastic processes by using graphical models. Section 2.3 provides the relevant related work on how other authors have approached solving similar problems with the first focus being on forecasting transaction volumes, then secondly looking at optimizing the spread of ATMs and lastly how influence modeling has been used for stochastic processes.

2.2 Reasoning Patterns in Bayesian Modeling

In the context of graphical models, influence flows in a specific way as demonstrated below on the three nodes with indirect connections; X,Y and Z. If we remove a node, this changes the influence relationships between these set of nodes. Take node Z for an example, the influence relationships to Z can flow from different directions as demonstrated in Figure 2.1 below.

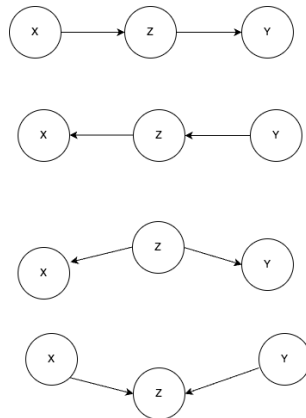


Figure 2.1: Different influence structures that depict the influence relationships of node Z with other nodes.

Given a Bayesian network structure as demonstrated in Figure 2.2, we know that if we remove node Z, in graphical models it is well understood that influence moves around through the network through the conditional independence associations.

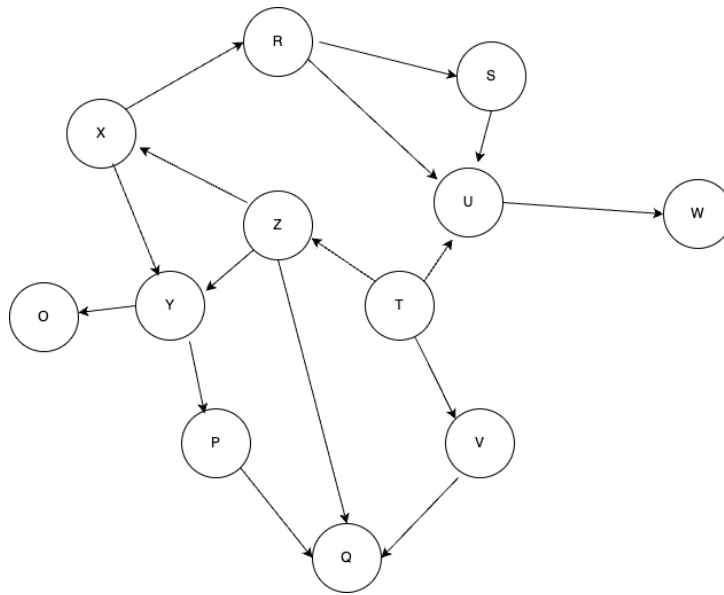


Figure 2.2: A Bayesian network influence structure

Figure 2.3 below depicts the new influence structure for when node Z is removed from the Bayesian network that is above. We can guarantee independence between variables only when there is no active trail between them. If we consider direct and indirect connections we can still have influence moving around in a graphical model structure. The only time where influence does not move around is when we have a V structure which blocks influence and this is the only exception. The reason influence does not move around in a V structure is because 2 nodes are conditionally independent of each other given the child node is not known.

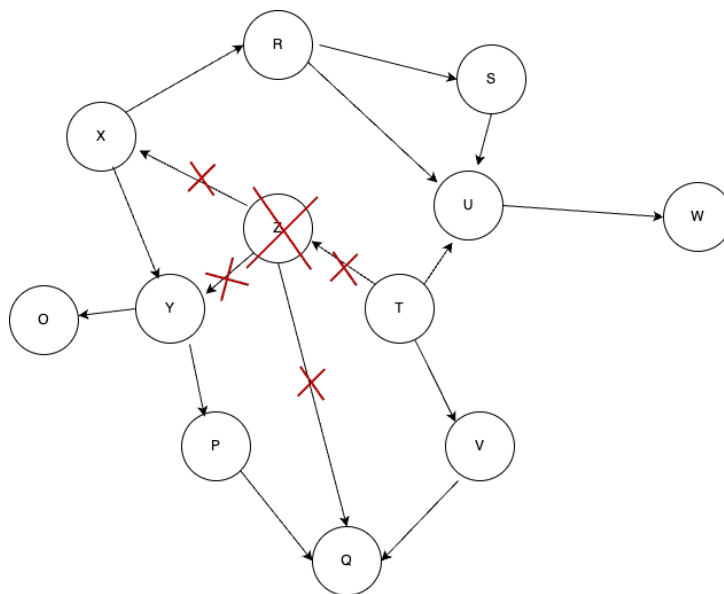


Figure 2.3: A Bayesian network influence structure with a removed node (Z).

2.3 Forecasting Techniques for ATM Networks

Accurate forecasting of ATM transactions is crucial for efficient financial planning and operational management. Various studies have explored time-series models, machine learning algorithms, and hybrid approaches to predict cash demand and transaction volumes. Traditional statistical models such as ARIMA and VARMAX have been widely used, while recent advancements leverage deep learning techniques like Long Short-Term Memory (LSTM) networks and convolutional neural networks for improved accuracy. The integration of feature engineering and clustering methods has further enhanced forecasting precision, allowing financial institutions to optimize cash replenishment schedules and reduce operational costs.

The Auto-regressive Integrated Moving Average (ARIMA) is a forecasting technique that was suggested by Box-Jenkins [Jenkins 1970] and this has enjoyed application in a variety of forecasting fields. The ARIMA uses time series data by predicting a value as a linear combination of its past values and past errors [Kamini *et al.* 2014]. Given a time series model with the form $ARIMA(p, d, q)$, the paper Kamini *et al.* [2014] proposed an ARIMA model, a Multi-Layer Feed Forward Neural Network (MLFF), Generalized Regression Neural Network (GRNN) and Wavelet Neural Network (WNN) to forecast cash demand in ATMs. In their study, Kamini *et al.* [2014] used NN5 competition data that contained daily time series data of cash withdrawals from 111 ATMs from different parts of England for a period of 2 years.

Firstly, Kamini *et al.* [2014] used traditional non-chaotic methods for forecasting where missing values were imputed with the data being deseasonalized with a time lag of 7 days. In the second method with non-traditional chaotic approaches, Kamini *et al.* [2014] reconstructed the phase space by using chaotic parameters, namely, embedding dimension and delay time using a TISEAN tool. The study used correlation dimension and the false nearest neighbor methods to identify chaos in the data. A sequential split was performed on the data for each of the ATM centers. The data was divided into 80% for training the 20% for testing. The best performing model when method 1 was used for forecasting was the GRNN with an SMAPE of 23.16%. In this method, the phase space is reconstructed using a delay time of 2 and an embedded dimension of 2, the GRNN also yielded the best Symmetric Mean Absolute Percentage Error (SMAPE) of 14.713%.

Rafi *et al.* [2020] proposed a time series model for predicting cash demand in a network of ATMs for a specific financial institution in Pakistan. Instead of using the standard ARIMA approach as done by Kamini *et al.* [2014], Rafi *et al.* [2020] built a Vector Auto Regressive Moving Average with Exogenous Variables model (VAR-MAX) by using the transaction data of each ATM. The study used transaction data of 7 ATMs from the period of June 2013 to December 2015 for a region with significant financial activity. The data was pre-processed for parsing into the model while the Dickey-Fuller test was used to verify the stationary test.

The split ratio was 70% for training and 30% for testing. The proposed model used by Rafi *et al.* [2020] puts into account the exogenous variables such as; holidays, salary week, and weekdays. The data was made stationary by utilizing differentiation techniques. The VARMAX model produced an average SMAPE of 31.4%, this is the same evaluation matrix that was utilized by Kamini *et al.* [2014] but it is significantly larger compared to their results and it must be noted that they used a different forecasting model and data.

Another prominent forecasting technique is the Long Term-Short Memory Networks (LSTMS) and these have been proven recently to outperform state of the art univariate time series forecasting methods [Bandara 2017]. Given a group of similar time series data, the authors Bandara [2017] forecast across time series databases using Long Term-Short Memory Networks. The LSTMS were used on two benchmark data sets, namely the CIF2016 and NN5 data sets. The CIF2016 dataset consisted of monthly time

series data composed of two subgroups, one group was related to the banking industry and the other was an artificially generated series. The forecasts were performed on a total of 72 time series for different months ahead forecasts.

The second dataset which was the NN5 data contained 2 years of daily cash withdrawals at various automated teller machines (ATMs) located in the UK. A total of 111 time series data for the ATMs was provided with the requirement that the forecasts must be performed for a prediction horizon of 56 days ahead. The NN5 data comprised of different real world forecasting challenges, such as outliers, step ahead forecasting, missing values, and multiple seasonalities.

It must be noted that it was the same data set that was used by Kamini *et al.* [2014] but a different model was used to solve for the same problem. The evaluation function that was used for both the CIF2016 and NN5 was the Symmetric Mean Absolute Percentage Error (SMAPE). Bandara [2017] proposed a new LSTM method which they called the LTSM.Cluster, and it yielded the following results for the datasets, for CIF2016 the SMAPE was 10.5% and for the NN5 dataset, the SMAPE was 21.6%. In the study by Kamini *et al.* [2014] the GRNN used in method 2 of their approach outperformed the LTSM.Cluster used by Bandara [2017] by a margin of 6.887%.

Andrawis *et al.* [2011] presented models to forecast 111 time series data which was the same data that was explored by Kamini *et al.* [2014] and Bandara [2017]. Andrawis *et al.* [2011] used time series data that span a period of 2 years with the aim of forecasting for a horizon of 56 days. The authors experimented with a combination of multiple forecasts and models for their study. Model pre-testing was done by firstly taking an important step to deseasonalize the time series data. The models were then pre-screened by making a comparison of computational intelligence and linear models.

A total of 140 models were experimented with and each of these models had different parameters, pre-screening and post-processing methods. The models were trained on 97 weeks of data and 8 weeks of data were used for testing. A total of 9 models turned out with the best accuracy, contrasting this study with Kamini *et al.* [2014], it used 4 more models for forecasting which made it a robust study. The combination of models were selected based on their accuracy. To evaluate model accuracy, the Symmetric Mean Absolute Percentage Error (SMAPE) was used, which is the main evaluation metric for the competition.

Similarly, Jörg [2011] used the NN5 time series data with hybrid models to forecast for a 56 day horizon. Their contribution to the NN5 competition was placed 6th out of a total of 19 teams. When the data was preprocessed it exhibited a weekly seasonality as evidenced on the autocorrelation function. A total of 3 different models were trained with the combination being a neural network ensemble, an ensemble of nearest trajectory models and lastly a model for the 7 day cycle. A weighted mean was calculated by combining all 3 models and the weights were inversely proportional to the forecast errors of the validation set. The evaluation metric used was the Symmetric Mean Absolute Percentage Error (SMAPE). The combined SMAPE for the validation set was 19.38%.

Coyle *et al.* [2010] utilized self-organizing neural networks on the same NN5 forecasting time series data that has been surveyed on the above literature. The self-organizing fuzzy neural network is a specific type of a fuzzy neural network [Leng and Prasad 2004]. The SOFNN is a neural network with five layers and can self-organize. Coyle *et al.* [2010] utilized structure learning for the learning process of the SOFNN and for parameter learning. The Symmetric Mean Absolute Percentage Error (SMAPE) was also used a measure to evaluate the SOFNN predictive outcomes, a mean SMAPE of 20.4% was achieved.

Rajwani *et al.* [2018] used exploratory data analysis and LSTMs for ATM cash flow prediction. ATM transaction data was collected for ten ATMs that were distributed in the Karachi region. The data

was collected for a period of 2 and a half years with 120 246 rows. A total of 60 features came with the data before being pre-processed. After performing exploratory data analysis and a few transformations of the data, Rajwani *et al.* [2018] found a few interesting observations about the data.

It was noticed that most transactions take place at the beginning of the month and at the end of the month. This can be attributed to the fact that people are most likely to be paid at the end or the start of the month. The split ratio for the experimentation was 70/30, with 70% for training and 30% for testing. This split ratio is similar to the ratio used by Rafi *et al.* [2020] in their study. The LSTM model had a split ratio of 67/33 and 20 epochs were ran. This yielded a MSE of 0.027 on the training data set and a MSE of 0.028 on the testing dataset. The RMSE indicated that they were 0.027 away from the actual target.

Simutus *et al.* [2007a] explored the use of neural networks and support vector machines to forecast cash demand for ATMs. Simutus *et al.* [2007a] used simulated and real data for training the models and evaluating their performance. The data was gathered or simulated for a period of 2-3 years for training a reliable ANN and SVR on a total of 15 ATMs. All ATM machines had a separate three layer feed-forward neural network with the input variables being coded values of weekday, day of the month, month of the year, holiday effect value and average daily cash demand.

The ANN was trained by utilizing the Levenberg-Marquardt optimization method with the output variable being the cash demand for the ATM for the next day. The data split ratio was 70% for training and 30% for testing. The MAPE of the ANN on the simulated data was 0.76% while for the SVR it was 4.1%. Since there is a total of 15 ATM sites, the authors took the average MAPE for both models and the following were the results; the average MAPE for all 15 ATMs when using the ANN model was 20.6% while for the SVR it was 25.1%.

Bhandari and Gill [2016] proposed an integrated technique for forecasting which uses the back-propagation algorithm with a genetic algorithm. In their research article, Simutus *et al.* [2007a] used ANNs and SVMs, in this article, Bhandari and Gill [2016] used a hybrid neural network for robustness. The proposed hybrid model that used BP/GA started with the collection of data, normalization, feature extraction, training and testing. A combination of 5 input variables were used and these were; day number, week day, weekend day, salary effect and holiday effect. Different population sizes were used with a different number of hidden neurons and iterations. The maximum population size achieved a Mean Absolute Error (MAPE) of 1.85%

In addressing the improvement of forecasting cash demand for ATMs, Arabani and Komleh [2018] used a Convolutional Neural Network on data that was provided by an Iranian banking institution. Daily ATM cash withdrawal data was used as input dataset for forecasting. Simutus *et al.* [2007a] utilized both methods for their study and they added a statistical method and CNNs. The statistical method starts with an F-test until it produces a regression model. The remaining methods split the data for training and testing, all four methods were evaluated by using the Root Mean Squared Error (RMSE). The CNN produced the best results from all 4 methods with an RMSE of 4.74% with cross validation.

Fallahfati *et al.* [2022] also studied the forecasting of ATM cash demand but in their case they looked at 2 scenarios, this being before and after the COVID 19 pandemic. Their aim was to accurately predict ATM cash demand by making use of statistical and machine learning models. The authors identified 3 categories for ATMs based on accessibility and environmental factors. The categories were namely; residential districts, business districts, and shopping and recreation districts. A 3 year period of daily withdrawal data from 2017 to 2020 was used. The 3 year time frame was split by the period in which the COVID-19 pandemic was observed. Parametric and non-parametric approaches were used for forecasting ATM cash demand. The authors used 5 parametric models and 4 non-parametric machine learning models. A total of 12 features were selected based on market dynamics in Tehran, Iran. To

evaluate the accuracy of their predictions, the authors used the Mean Squared Error (MSE). The best performing model was the regression Random Forest (RF) approach with an MSE of 0.0101.

Simutus *et al.* [2007a] explored the use of neural networks and support vector machines and similarly, Ramírez and Acuña [2011] also made use of ANNs and SVMs. The authors utilized the NN5 competition data with a focus on daily cash withdrawals for ATMs in England. A total of 691 data points were used for training from 30 ATMs, and 100 data points were used for testing purposes. The standard evaluation metric for the NN5 competition is the SMAPE and the authors utilized the same metric to evaluate their models. The best performing models were the MLP with a mean SMAPE of 21.37%.

Gurgul and Suder [2016] studied the calendar and seasonal effects on daily ATM withdrawals for ATMs that are managed by the Euronet network. The study used was based on ATM data for two regions and these were namely, Lesser Poland and Subcarpathia. A total of 222 ATMs were used for a period starting from 2010 January to 2012 December. The authors were able to prove the existence of calendar effects on ATM withdrawals. Different days exhibited different withdrawals and this dependence was on the day of the week, week of the month or the month of the year.

Previous authors that are surveyed above mostly use machine learning approaches for forecasting ATM cash demand, Venkatesh *et al.* [2013] proposed the use of neural networks with the clustering of ATMs by grouping them according to features. The ATMs were clustered by making use of the Levenshtein SAM distances. The authors made use of the NN5 cash withdrawal data for a period of 2 years. A total of four neural network architectures were used and these were; Multi-Layer Feed Forward (MLFF), Wavelet Neural Networks (WNN), Group Method of Data Handling (GMDH) and the Generalised Regression Neural Network (GRNN). The GRNN achieved the the best SMAPE of 23.16% and the overall approached utilized by the authors outperformed the results of Andrawis *et al.* [2011].

Sarveswararao *et al.* [2022] studied ATM cash demand for an Indian bank by modeling chaos and making use of deep learning. A total of 50 ATMs were used for the study and 28 of the ATMs were observed to exhibit chaos and the 22 remaining ATMs did not have chaos observed in their data. A combination of a statistical technique with machine learning techniques that ranged from deep learning techniques that are Long Short Term Memory (LSTM) neural networks. A total of 12 different techniques were utilized for forecasting ATM cash demand. The deep learning LSTM model outperformed all other models and it achieved a mean SMAPE of 24.17%.

As a predecessor to the work done by Simutus *et al.* [2007a], Simutus *et al.* [2007b] explored the use of a flexible neural network for forecasting ATM cash demand. The ANN was also trained by making use of the Levenberg-Marquard optimization method and the data was split to a ratio of 70% for training and 30% for testing. Simulation data was generated to depict the behavior of a typical ATM in Kaunas city, in Lithuania. The Mean Absolute Percentage Errors (MAPE) values varied between 15-20% for daily predictions and they varied between 8-15% for weekly predictions.

Suder *et al.* [2007] studied the effectiveness of forecasting methods for ATM withdrawals under different market conditions. Suder *et al.* [2007] also evaluated whether the pandemic-driven market conditions had an impact on the models for predicting withdrawals similarly to what Fallahfati *et al.* [2022] explored in their study. The data was gathered from one of the largest ATM operators in Poland for a total of 61 ATMs that consisted of daily withdrawal data for a period of 4 years. A combination of statistical and machine learning methods were used for forecasting ATM cash withdrawals. The XGBoost technique outperformed all the methods that were tested and it produced a combined mean SMAPE of 37.92% for both pre and post the COVID-19 pandemic.

ATM demand forecasting focuses on the daily demand of each ATM, and not on the denominations

dispensed [du Toit 2011]. A total of 18 ATMs were used to conduct the study on forecasting the demand for denominations. A client normally withdraws an amount at an ATM and receives different denominations that make up the amount. du Toit [2011] focused on forecasting the aggregated denominations instead of forecasting the demand for each denomination. The ATM data was in the form of a time series and it exhibited multiple seasonal patterns of different lengths. The empirical daily dispensing data was collected for a period of 3 years, from January 2007 to November 2009. du Toit [2011] employed a variety of forecasting techniques, namely; regression, decomposition, exponential smoothing and Box-Jenkins methods.

du Toit [2011] used the Mean Absolute Percentage Error (MAPE) and the results for each technique were tabulated for comparison. The author developed models for daily, weekly and monthly forecasts. The daily forecasts saw an improvement of 21.67% with the MAPE being 58.01%. The new weekly forecast had a MAPE of 10.55% with an improvement of 18.91% from the previous forecasting technique. Lastly, the MAPE for the monthly forecasts was 3.17% with an improvement of 42.36%.

Keyter [2004] used regression analysis to forecast transaction volumes on ATM devices for a South African bank. A variety of variables were taken into account before regression was performed, and this is due to the fact that these variables influence the transaction volumes differently. A final sample of 86 ATM sites was taken into account out of a population size of 1800 ATM sites. A different combination of the features out of a total of 18 features were tested, this was done until the combination producing the highest R^2 was found. ATMs can be located at different sites namely, shopping center, industrial areas, garages, convenience stores and business nodes. The results for the shopping centers used 3 features and the average residual was 29% with $R^2 = 0.85$.

Table 2.1: A table comparing different forecasting techniques that were used by other authors.

Author	Data	Models	Accuracy
[Kamini <i>et al.</i> 2014]	111 Time series data	GRNN	SMAPE= 14.713
[Bandara 2017]	111 Time series data	LSTM.Cluster	SMAPE = 21.6
[Andrewis <i>et al.</i> 2011]	111 Time series data	Combined (all 9 models)	SMAPE = 18.95
[Rafi <i>et al.</i> 2020]	Transaction data	VARMAX	SMAPE= 31.4
[Keyter 2004]	Data for 86 ATMs	Regression Analysis	$R^2 = 0.85$
[Rajwani <i>et al.</i> 2018]	Transaction data	LSTM	MSE= 0.028
[Simutus <i>et al.</i> 2007a]	Transaction data	ANN	MAPE= 20.6
[Simutus <i>et al.</i> 2007a]	Transaction data	SVR	MAPE= 25.1
[du Toit 2011]	Denomination data	Fuzzy ARIMA	MAPE= 3.17
[Jörg 2011]	111 Time series data	Hybrid models	SMAPE= 19.38
[Coyle <i>et al.</i> 2010]	111 Time series data	SOFNN	SMAPE= 20.4
[Bhandari and Gill 2016]	111 Time series data	Hybrid ANN	MAPE= 1.85
[Arabani and Komleh 2018]	Transaction data	CNN	RMSE= 4.74
[Fallahfati <i>et al.</i> 2022]	Transaction data	Regression RF	MSE= 0.0101
[Ramírez and Acuña 2011]	Transaction data	MLP	SMAPE= 21.37
[Venkatesh <i>et al.</i> 2013]	Transaction data	GRNN	SMAPE= 23.16
[Sarveswararao <i>et al.</i> 2022]	Transaction data	DL LSTM	SMAPE= 24.17
[Simutus <i>et al.</i> 2007b]	Transaction data	ANN	MAPE= 14 – 20
[Suder <i>et al.</i> 2007]	Transaction data	XGBoost	SMAPE= 37.92

2.4 Optimizing Cash Flow and Resource Allocation

Optimizing cash flow and resource allocation in ATM networks is essential because banks must balance the need to minimize costly operations such as cash transportation, insurance, and cash freezing costs with the imperative of ensuring that ATMs remain adequately stocked to prevent service disruptions and customer dissatisfaction. This presents a Multi-Objective optimization problem, a multi-objective optimization problem (MOP) can be defined as an optimization problem with multiple objectives measured with different performance functions, usually in conflict with each other, and a set of restrictions [Rodriguez and Lozano 2008]. Li *et al.* [2020] proposed an end to end framework for solving multi-objective optimization problems using deep reinforcement learning(DRL) and termed it DRL-MOA. The MOP are decomposed into scalar optimization sub-problems and then each problem is modeled as a neural network. The Pareto optimal solutions were obtained through the trained neural network models. The commonly known multi-objective traveling salesman problem was solved by using (DRL-MOA) by modeling the sub-problem as a Pointer Network.

Armenise *et al.* [2012] investigated the optimization of ATM cash by using a genetic algorithm in order to produce optimal load strategies. A pool of 30 ATMs was used as a test bench, and the features were; location, position, cash capacity, and usage. Similarly to how we want to reduce financial costs, the paper focused on reducing the costs of unused stock in ATMs. The implemented algorithm made use of JENES which is an open source Java library for genetic algorithms. The standard deviation, average and the p-value were used as evaluation functions for the various combinations, and the results were tabulated for each for the study.

Cisternas-Vera *et al.* [2016] looked at the impact of mobile banking on reshaping the bank branch network for an American (US)-based bank. Taking the geo-coded transaction data, a dynamic structural model was developed and this represented the consumers' preferences for online and physical channels. The model was used to optimize the branch network in terms of capacity, amenities, location, and number of branches. The data comprised a sample for more than 500 000 accounts with more than 1.7 billion transactions across all channels from June 2007 to June 2015. A combination of discrete choice dynamic programming and linear programming was used to develop the model. The authors in the above study solved the problem of optimization using various techniques.

Bilir and Doseyen [2018] focused on the optimization of ATM and branch cash operations using an integrated cash requirement forecasting and cash optimization model. The aim of the study was to optimize the cash supply chain for ATMs and branches for a mid-sized bank in Turkey. In optimizing the cash supply chain, the main objective of the model was to minimize idle cash levels without decreasing customer satisfaction. The optimization phase of the study formulated an optimum transfer schedule for branches and ATMs by making use of an integer formulation. The aim of the optimization model was to minimize total cost, which had two parts, transfer costs and the penalty fee for maintaining idle cash.

Broda *et al.* [2014] presented an approach for optimizing fill-in strategies for ATMs for a bank in Serbia. The Credit Agricole Bank had an aim of minimizing costs which were listed as; cash freezing costs, transportation costs and insurance costs while ensuring customer service was not impacted. The data used were daily transactions for a total of 40 ATMs in the region of Novi Sad. The approach used by the authors was to firstly model daily cash demand by using a combination of time series analysis, linear regression and the compound poisson process. Lastly, the authors then optimized the ATM filling strategies by using a simple algorithm that performs an exhaustive search for a feasible solution space.

Ekinci *et al.* [2014] attempted to optimize the ATM cash replenishment strategies by utilizing group-demand forecasts. The data for a total of 152 ATMs in Istanbul was used for a period of 2 years and 6 months. Based on the longitude and latitude of each ATM, these were grouped via an equation which generated a total of 27 groups. Cash withdrawals were averaged daily for each group and forecasting

techniques were then applied for each group. An optimal time between two replenishments was found by using an objective function with constraints.

Nigmatuli and Chaganva [2022] created a cash management system for ATMs and information payment terminals. The data was sourced from Kaggle which is open source and it represented cash withdrawals for a period of 6 years. The study was conducted on a total of 3 ATMs in India, Bangalore. A combination of forecasting models were used to sufficiently obtain a good level of correct forecasts. The triple exponential model, singular spectrum analysis model, and a recurrent neural network with Long Short-Term Memory were used for forecasts. A 2 week forecast horizon was used for each ATM and the discrete optimal control was used for optimization. Similarly, Lazaro *et al.* [2017] attempted to improve cash logistics for bank branches for their case study but they used a similar approach that was utilized by Nigmatuli and Chaganva [2022]. Lazaro *et al.* [2017] attempted to optimize cash logistics by firstly forecasting transactions by using machine learning and applying a robust optimization model. The data was sourced from a national bank for a period of 4 years. The cash demands that were generated were then fed onto a defined integer programming problem for optimization.

Dilijonas *et al.* studied the problems of management and optimization of ATM network systems. A flexible artificial neural network approach was used for cash demand forecasting. Each machine was assigned a separate 3 layer feed-forward neural network and these were trained based on the Levenberg-Marquardt optimization method. A combination of multi-agent technologies and ANNs were used for optimization. The authors found that a combination of the technologies gave an advantage for managing cash optimization dynamically in complex systems. The model resulted in 20-30% average reduction in cash.

Serengil and Ozpinar [2019] much like Nigmatuli and Chaganva [2022] and Lazaro *et al.* [2017] utilized cash flow prediction with a combination of an optimization technique. The data was sourced from a Turkish bank for a total of 6500 ATMs. The data span a period of years starting from 2013. An artificial neural network was used to forecast the next day's demands with a total of 15 features. The cash demand forecast from the neural network model were then consumed to compute the optimal replenishment strategy using a cost function that used linear optimization.

Thanh *et al.* [2023] viewed the ATM cash replenishment problem as a multi-objective logistics optimization problem. The authors extended the Vehicle Routing Problem to formulate a Multi-objective Optimization Problem (MOP). The authors stated the problem as a mathematical problem which became a complete directed graph with vertices. Constraints were then applied to the model and these needed to be satisfied in order to come up with an optimal load strategy. The OR-Tools package was used for optimization by initializing variables based on the constraint of the MOP. The testing and evaluation of the model was done by using data from a bank in Hanoi, Vietnam.

Van Anholt and Vis [2010] proposed a novel dynamic approach that factors in both ATM cash demand forecasting and ATM replenishment in an integrated way. The authors proposed decomposition on the forecasting component of the integrative approach. On this case decomposition was a changed linear regression technique, based on the demonstrated results, the decomposition outperformed the artificial neural network approach. Ghodrati *et al.* [2013] studied the ATM cash management problem by utilizing a genetic algorithm. They sourced their data from an Iranian bank for a period of 2 years for 164 ATMs. The ATMs were clustered into 3 groups of high, medium and low. A genetic simulation optimization algorithm was implemented as a solution.

Chotayakul *et al.* [2013] formulated the cash inventory problem for ATMs as a Mixed Integer Program (MIP). They developed it into the path formulation in deriving the near-optimal solution of the problem. The ATM network that was studied came from a bank based in Thailand. Bati and Gozupek [2017] devised an integer linear program (ILP) that was able to jointly optimize routing and cash man-

agement for a new generation of ATMs. Their aim was to minimize costs like idle cash costs and costs associated with logistics. A polynomial-time heuristic algorithm was implemented by using CPLEX.

Kurdal and Sebestyenova [2013] viewed the routing optimization problem as a dynamic vehicle routing problem (VRP) which is similar to what was explored by Thanh *et al.* [2023]. Kurdal and Sebestyenova [2013] solved the formulated problem by using a parallel genetic algorithm. Kiyaei and Kiaee [2021] studied ATM cash replenishment planning by making use of a deep Q-network for optimization. This model used a combination of deep learning and Q-learning. In their study, Kiyaei and Kiaee [2021] came up with two components for solving the problem. The first component learns the dynamics of the cash demand while the second one uses Q-learning to learn the action-value function.

Mubiru and Nalubowa Ssempijja [2024] attempted to model and optimize cash load strategies for ATMs under stochastic demand. A total of ten ATMs were used for the study for commercial hubs in the city of Kampala. Each ATM's data was based on hourly transactions for a period of 4 months. The authors were able to develop an optimization model which used the Markov decision approach for cash loading strategies. Li *et al.* [2009] tackled a unique problem of selecting the best site for a new ATM by using the Particle Swarm Optimization algorithm. Their study focused on a bank based in China, Tianjin, and they surveyed a total of 50 ATMs. In their preliminary results, Li *et al.* [2009] demonstrated that they can address the site selection problem while at the same time being able to forecast transactions for the same site.

Table 2.2: A table comparing different optimization techniques that were used by other authors.

Author	Data	Models
[Armenise <i>et al.</i> 2012]	Transaction data	Genetic Algorithm
[Cisternas-Vera <i>et al.</i> 2016]	Transaction data	Linear and dynamic programming
[Bilir and Doseyen 2018]	Transaction data	Integrated model
[Broda <i>et al.</i> 2014]	Transaction data	Exhaustive search
[Ekinci <i>et al.</i> 2014]	Transaction data	Linear programming model
[Nigmatuli and Chaganva 2022]	Transaction data	Optimal control
[Lazaro <i>et al.</i> 2017]	Transaction data	Integer programming
[Dilijonas <i>et al.</i>]	Transaction data	Multi-agent technologies
[Serengil and Ozpinar 2019]	Transaction data	Cost function and linear optimization
[Thanh <i>et al.</i> 2023]	Transaction data	Pareto-based multiobjective learning algorithms
[Van Anholt and Vis 2010]	Transaction data	Decomposition model
[Ghodrati <i>et al.</i> 2013]	Transaction data	Genetic Algorithm
[Chotayakul <i>et al.</i> 2013]	Transaction data	Mixed Integer Program (MIP)
[Bati and Gozupek 2017]	Transaction data	Heuristic CPLEX
[Kurdal and Sebestyenova 2013]	Transaction data	Parallel Genetic Algorithm
[Bati and Gozupek 2017]	Transaction data	Q-learning
[Mubiru and Nalubowa Ssempijja 2024]	Transaction data	Markov decision approach
[Li <i>et al.</i> 2009]	Transaction data	Particle Swarm Optimization

2.5 Related Work

2.5.1 Mapping Influence in Complex Systems

Understanding influence relationships in probabilistic graphical models is essential for analyzing the behavior of interconnected systems. Influence modeling within Bayesian networks has been applied across domains to capture dependencies and quantify structural changes. Research in this area primarily employs structure learning techniques, including score-based and constraint-based methods, to recover underlying influence patterns from data. These approaches have been used in diverse applications, such as modeling stochastic processes, evaluating resilience in complex networks, and optimizing decision-making frameworks. By leveraging probabilistic reasoning, influence modeling provides valuable insights into how system dynamics evolve under various conditions, including node removal and disruptions.

Influence diagrams and belief networks are defined as directed graphical models for representing models of probabilistic reasoning and decision making [Shachter 2007]. Important relationships can be captured easily with the graphical representation of influence diagrams and belief networks. Influence diagrams are more suited to capture the conceptual relationship between inference and decision making [Biedermann and Taroni 2023]. Influence networks defined as Bayesian networks whose probabilities are approximated by expert provided influence constants [Zaidi *et al.* 2010]. Influence relations in graphical models refers to the dependencies or causal connections between variables represented within a graph [Koller and Friedman 2009].

Ajoodha [2018] attempted to recover the influence relations by using a complete algorithm for a collection of partially observable multi-dimensional stochastic processes. Score-based structure learning was used to identify the optimal structure by applying a scoring metric to evaluate structural configurations in relation to the data. Ajoodha [2018] expands on score-based structure learning concepts for tracking influence among stochastic processes.

Firstly, Ajoodha [2018] decomposed the distributions represented by stochastic processes into a series of temporal models. Then, an assembly and a scoring function was defined to assess the quality of potential influence networks. This then became a combinatorial optimization problem, which required the navigation of the search space to find the optimal influence network, the best fitting structure for the training data was then returned. Ajoodha and Rosman [2020] attempted to study the influence between stochastic processes that are represented by hidden Markov Models (HMMs). A structure learning approach was also used in this study and the candidate influence structures were dynamic influence networks (DINs) which model the relationships between the stochastic processes.

In a different study, Ajoodha and Rosman [2021] used constraint-based structure learning to recover the direct influence between hierarchical dynamic Bayesian networks. This study still used structure learning but with focus being on hierarchical dynamic bayesian networks. The learning tasks were compared by using KL-divergence. In another study, Ajoodha and Rosman [2017] still made use of score-based structure learning in an attempt to recover the influence structure but in this case naïve Bayes models were used instead of hidden Markov Models or hierarchical dynamic Bayesian networks as stated before. A total of four parameter or structure learning tasks were explored to recover the ground truth distribution.

Mohammadi and Wit [2014] concentrate on Bayesian structure learning in Gaussian graphical model, addressing both decomposable and non-decomposable cases. They introduced a new Bayesian framework for determining Gaussian graphical models and developed a birth-death Markov chain Monte Carlo (BDMCMC) algorithm to carry out both structure learning and parameter learning. To evaluate the performance of the BDMCMC algorithm, the KL-divergence was used on the precision matrix.

Chen *et al.* [2020] introduced a Bayesian network-based approach to model Nitrogen utilization efficiency of dairy cows. Historically, previous authors utilized statistical models to predict manure N excretion. Chen *et al.* [2020] used a dataset from the digestibility studies from the AGri-Food and Biosciences Institute in Northern Ireland. The Growth-Search as a constraint-based structure learning approach was used to recover the influence between the variables. The Bayesian network model was able to capture the relationships between the variables and it also established a causal influence model. In another study, Zheng *et al.* [2016] used Bayesian networks to model methane emissions from milking dairy cows. Previously, different regression based models were applied to examine the findings which is similar to how Chen *et al.* [2020] extended from statistical models to Bayesian networks. The dataset was also collected from the Agri-Food and Biosciences Institute in Northern Ireland. The evaluation took place on dataset with 934 milking dairy cows. Structure learning was done by using Hill-Climbing structure learning and it utilized a score-based approach.

Matejová *et al.* [2023] used influence modeling to study the factors associated with atherosclerosis. Cardiovascular diseases have many factors that are associated with atherosclerosis [Matejová *et al.* 2023]. A combination of statistical methods and different machine learning techniques were utilized for the study. The Bayesian networks in this instance were used to study the relationships between the different factors.

Tang *et al.* [2020] evaluated the resilience of urban transportation systems by utilizing a Bayesian network model. The study focused on investigating the resilience of road transportation for 4 cities in China. These cities were Beijing, Tianjin, Shanghai and Chongqing for a period starting in 1998 to 2017. A hierarchical Bayesian network model was proposed to evaluate the resilience of these systems. Backward inference, forward inference and sensitivity analysis were used to test the proposed model and the results demonstrated reasonable values. Hosseini *et al.* [2016] assessed system resilience by also utilizing Bayesian networks similar to the study done by Tang *et al.* [2020]. Hosseini *et al.* [2016] focused on a case study of a sulfuric acid manufacturing plant that had 2 suppliers of sulfur. The developed Bayesian network model was able to capture various disruptions ranging from power outages, technical errors to hurricanes. Hosseini *et al.* [2016] validated their model by making use of sensitivity analysis which is the same approach that was used by Tang *et al.* [2020] for evaluating their model. Special types of reasoning were performed and these were belief propagation for inference purposes.

Jaradat *et al.* [2020] modeled the inter-dependencies between critical features by making use of a Bayesian network. Their case study was focused on an inland waterway port and surrounding supply chain network. A Bayesian network captured the interdependency between the factors. The results were probed by making use of belief propagation and sensitivity analysis. Hossain *et al.* [2019] evaluated another study for an interdependent electrical infrastructure system. This study also used a Bayesian framework to assess system resilience like in Jaradat *et al.* [2020]. The framework addressed a range of possible disruptions and proposed solutions to mitigate their consequences. A case study of the Washington, DC electrical infrastructure system was used. To evaluate resilience, different measures were used and these were sensitivity analysis, forward propagation, information theory and backward propagation.

Liu *et al.* [2020] presented a robust dynamic Bayesian network method for supply chain disruption risk estimation. The authors proposed a new and general nonconvex programming model, and this was compared with the classic DBN approach on a case study. Kammouh *et al.* [2020] evaluated the resilience of engineering systems by utilizing a probabilistic framework. The authors developed 2 approaches that could handle static and dynamic systems. A Bayesian Network (BN) was used for static engineering systems and Dynamic Bayesian Networks (DBN) were used for dynamic systems. The paper made use of 2 illustrative systems to apply the methodology on, one being evaluating resilience for a transportation system and the other being for Brazil.

Table 2.3: A table comparing different influence modeling techniques that were used by other authors.

Author	Models for mapping influence
[Ajoodha 2018]	Score-based structure learning
[Ajoodha and Rosman 2020]	Hidden Markov Models
[Ajoodha and Rosman 2021]	Constraint-based structure learning (hierarchical dynamic Bayesian networks)
[Ajoodha and Rosman 2017]	Score-based structure learning
[Mohammadi and Wit 2014]	Birth-death Markov chain Monte Carlo (BDMCMC)
[Chen <i>et al.</i> 2020]	Growth-Search
[Zheng <i>et al.</i> 2016]	Bayesian modeling
[Matejová <i>et al.</i> 2023]	Bayesian modeling
[Tang <i>et al.</i> 2020]	Bayesian modeling
[Hosseini <i>et al.</i> 2016]	Bayesian modeling
[Jaradat <i>et al.</i> 2020]	Bayesian modeling
[Hossain <i>et al.</i> 2019]	Bayesian modeling
[Liu <i>et al.</i> 2020]	Dynamic Bayesian network
[Kammouh <i>et al.</i> 2020]	Bayesian network and dynamic Bayesian network

2.5.2 Conclusion

The related work explored in this chapter spans multiple domains, including forecasting ATM transaction volumes, optimizing cash flow and resource allocation, and mapping influence in complex systems. Each of these areas provides crucial insights that contribute to understanding ATM networks as probabilistic graphical models. Forecasting techniques have demonstrated various approaches to predicting cash demand and transaction volumes, offering methods that aid in resource planning and operational efficiency. Time-series models, machine learning algorithms, and hybrid forecasting techniques have been extensively studied to improve accuracy and adaptability under varying market conditions. These methodologies set a foundation for understanding transaction patterns, which is essential when analyzing network resilience following disruptions. Optimization studies have focused on the logistical and financial challenges of ATM networks, emphasizing how resource allocation strategies can minimize operational costs while maintaining service availability. Various optimization techniques, including mixed-integer programming, heuristic algorithms, and simulation-based approaches, have been employed to enhance cash replenishment scheduling and ATM placement decisions. These strategies highlight the significance of balancing cash flow efficiency with customer demand to ensure uninterrupted service.

The discussion on influence modeling extends these considerations by examining how interdependencies between network nodes impact overall system behavior. Studies in this area provide a framework for understanding the propagation of disruptions across networks and how probabilistic graphical models, such as Bayesian networks, can be employed to analyze these interactions. Research on network resilience, influence maximization, and cascading failures in complex systems further contextualizes the importance of modeling ATM networks as interconnected structures subject to perturbations. By integrating these themes, this chapter establishes a comprehensive foundation for the research problem addressed in this dissertation. The intersection of forecasting, optimization, and influence modeling provides a structured approach to investigating how ATM networks respond to node removal. The insights gained from prior work inform the methodology adopted in subsequent chapters, where the impact of node removal on Bayesian networks is analyzed. This study contributes to the broader research landscape by applying Bayesian learning techniques to quantify structural and probabilistic changes in ATM networks, bridging the gap between theoretical modeling and real-world financial systems.

Chapter 3

Methodology

This chapter outlines the methodological approach for analyzing the impact of node removal on Bayesian networks representing ATM networks. Our study aims to develop a comprehensive framework for understanding how the removal of nodes which represent ATMs affects the overall network structure and its probabilistic dependencies. We begin by formulating the problem mathematically, modeling the ATM network as a directed acyclic graph (DAG) with conditional probability distributions (CPDs) assigned to each node. This formulation enables us to capture the complex interactions and dependencies between transaction volumes at different ATMs.

We then present the research objectives, accompanied by a detailed experimental design to investigate the effects of node removal on the joint probability distribution and the recovered network structure. Our methodology incorporates structure learning and parameter estimation techniques to recover the network post-removal, and Kullback-Leibler (KL) divergence to quantify deviations from the original distribution. By modeling these dynamics, we aim to provide insights into the resilience and stability of ATM networks under varying disruption scenarios. This approach contributes to the broader understanding of influence relationships in probabilistic graphical models, particularly in applications requiring robust and adaptive network architectures.

3.1 Problem Formulation

Many problems in the field of science can be modeled as stochastic processes. Stochastic processes are used to describe the evolution of random variables over time or space. Applications of stochastic processes can be used for modeling and analyzing random phenomena, giving a clear understanding into how the complex systems interact under uncertainty. We can learn the influence between these variables if we know that they interact with each other. Suppose we have a set of variables $\mathcal{X} = \{X_1, \dots, X_n\}$ and we have a joint distribution $P_{\mathcal{G}}(X | Y)$ that represents the relationship between them. A Bayesian network that represents our problem as a tuple $\mathcal{B} = (\mathcal{G}, \Theta)$ can be generated where $\mathcal{G}$ is a directed acyclic graph (DAG) whose nodes represent the set of variables from above $\mathcal{X} = \{X_1, \dots, X_n\}$ and Θ represents the parameters of the probability distributions associated with each node in the graph. These parameters define how likely certain values are for a given node, conditioned on its parents in the graph.

Suppose that a subset of the variables in this set $\mathcal{X} = \{X_1, \dots, X_n\}$ had to be mutilated, how would this affect the joint probability distribution $P_{\mathcal{G}}(X | Y)$ that represented the relationship between these variables. Firstly, we will attempt to learn the parameters of a joint probability distribution which describes the relationship between the variables $\mathcal{X} = \{X_1, \dots, X_n\}$. We will then attempt to learn parameters of a probability distribution which describes the relationship between the variables $\mathcal{X} =$

$\{X_1, \dots, X_n\}$ when a subset of these variables is removed. A probabilistic graphical model can be used to represent the dependencies between different variables, with this graphical model we can then perform inference and learning for knowledge discovery.

3.2 Research Questions and Experimental Setup

Based on our problem formulation, we looked at three research questions:

Research Question 1: What is the impact on the KL divergence when nodes are removed from a recovered Bayesian network?

To address this question, we systematically mutilated Bayesian networks by removing nodes and the edges connected to them. For each modified network, we performed structure learning using a score-based approach, estimated the CPDs with Bayesian estimation, and compared the recovered network to the original ground truth distribution. The KL divergence served as the primary metric for quantifying the deviation between the original and recovered distributions. By experimenting with varying percentages of node removal and analyzing the results, we identified patterns and thresholds where network recovery begins to degrade significantly.

Research Question 2: What are the implications of varying structural complexity on the recovery of Bayesian networks and their KL divergence?

We explored this question by generating synthetic Bayesian networks with distinct structural configurations, including tree-like, sparse, and dense structures. For each structure type, we evaluated the performance of structure learning algorithms under different sample sizes and levels of node removal. By comparing the KL divergence across these configurations, we assessed how structural complexity influences the network’s recoverability. This analysis highlighted the trade-offs between complexity and robustness, providing insights into the resilience of different network designs under disruptions.

Research Question 3: How can insights from Bayesian networks improve the operational efficiency and planning of ATM networks?

To tackle this question, we modeled ATM networks as Bayesian networks, where nodes represented individual ATMs, and edges captured dependencies between their transaction volumes. Using historical transaction data, we trained and validated these models, then we simulated disruptions (e.g., ATM outages) by removing nodes. By analyzing the resultant changes in network influence and identifying critical nodes or clusters, we derived actionable insights for improving operational efficiency. This includes optimizing cash replenishment schedules, predicting the impact of outages, and planning the deployment of new ATMs to enhance network resilience and customer satisfaction.

3.2.1 Experimental Setup

In this study, we aimed to investigate the effect of removing a subset of variables from this set $\mathcal{X} = \{X_1, \dots, X_n\}$ on the joint probability distribution of the graph which can be represented as $P_{\mathcal{G}}(X | Y)$. We modeled the relationship between the variables using a Bayesian network as a tuple $\mathcal{B} = (\mathcal{G}, \Theta)$ where $\mathcal{G}$ is a directed acyclic graph (DAG) whose nodes represent the set of variables from above $\mathcal{X} = \{X_1, \dots, X_n\}$. We generated a random ground truth distribution for this Bayesian network by creating a Bayesian structure and assigning conditional probability distributions (CPDs) to each node in the network.

The structure of the Bayesian network was generated by randomly specifying the number of nodes and randomly connecting them with edges to form a directed acyclic graph (DAG). Each node in the

network had a CPD assigned based on its parents in the graph. We generated the CPDs randomly by using a distribution that best described the nature of our variables and met the assumptions of our model. We chose from a variety of distributions, this being from a Bernoulli distribution, Gaussian distribution, Dirichlet distribution and we ended up using the Bernoulli distribution which was a specific distribution which best met our assumptions of our model and described the nature of our chosen variables.

When the structure of the Bayesian network and the CPDs were defined, we generated random samples from the Bayesian network to create a dataset. Suppose that the learned probability distribution from structure learning of our Bayesian network that we generated was represented by $P_G(X | Y)$, and suppose that we remove a node or a subset of nodes from the graph and we learn the new probability distribution as $P_{\hat{G}}(X | Y)$. We can make use of our generated dataset from the random samples to perform probabilistic queries that will help us gauge how removing a node or a subset of nodes influences other nodes in the network.

We choose Bernoulli conditional probability distributions (CPDs) for each node because, after quantizing ATM transaction volumes into binary states (e.g., “high-volume” vs. “low-volume” activity), the presence or absence of significant transaction patterns is most naturally modeled as a two-state random variable. Bernoulli CPDs thereby allow us to capture essential fluctuations in ATM usage while maintaining tractable inference: exact likelihood calculations and structure-learning algorithms (hill-climb, tree search) remain computationally efficient even for networks of hundreds of nodes. Furthermore, focusing on binary CPDs simplifies the interpretation of KL-divergence results—any change in node behavior manifests directly as shifts in binary probabilities—without confounding effects from multi-state parameterizations. We acknowledge that real ATM transaction data often exhibits richer, continuous or count-based dynamics; in future extensions, Gaussian or Poisson CPDs could be incorporated once sufficient real-world data is available, but the Bernoulli framework provides a robust proof-of-concept for analyzing network resilience under node removal.

3.3 Application Domain

Each variable can be represented as a node in a graphical model, with associated variables being the parameters of each node. We have now represented the problem as a probabilistic graphical modeling problem, firstly we will represent the problem statement as a graphical model, generate random Bayesian networks for the true distribution, and show how inference can be done on the model for knowledge discovery.

3.3.1 Modeling Influence Relationships

Suppose we have this set of variables $\mathcal{X} = \{X_1, \dots, X_n\}$ and we have a joint distribution that represents the relationship between them. We can formally define a Bayesian network that represents our problem as a tuple $\mathcal{B} = (\mathcal{G}, \Theta)$ where:

- $\mathcal{G}$ is a directed acyclic graph (DAG) whose nodes represent the set of variables from above $\{X_1, \dots, X_n\}$.
- Θ is a set of parameters that specify the conditional probabilities $\Theta_{\mathcal{X}_i} | Pa(X_i)$ for each node X_i given its parent $Pa(X_i)$ in $\mathcal{G}$.

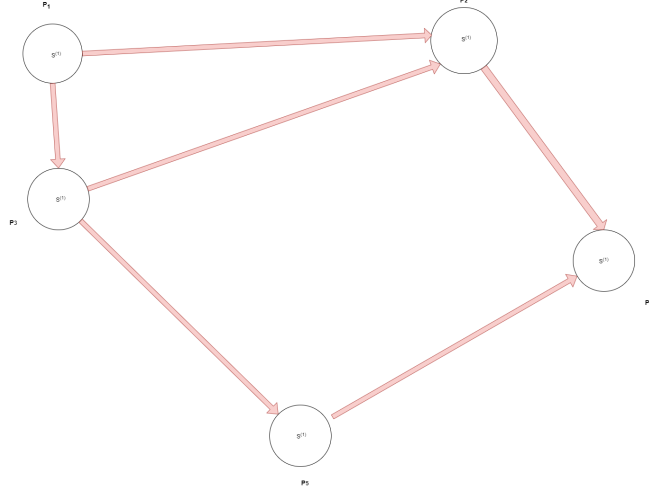


Figure 3.1: An influence structure between stochastic processes. The red arrows indicate the influence between these processes while the solid arrows indicate the parameters of each node.

Let P_1 in Figure 3.1 represent a node in the network which is made up of n different nodes. If we assume that the future is conditionally independent of the past given the present, this system would maintain the Markov property allowing us to make the Markov assumption. If the structure or the parameters of the system do not change under any transformations, this would maintain the invariance assumption of the system allowing us to make the stationary assumption.

Our ground truth distribution was generated by following the process that we touched on in the experimental set up section. We firstly defined the graph structures by choosing to use between 50-100 variables that were on the nodes in our graph structure, which was a directed acyclic graph (DAG) that is represented by Figure 3.1. We explored tree-like, sparse and dense structures for our ground truth distributions. We generated random Bernoulli distributions that best described our graph in Figure 3.1. These distributions specified the conditional probability distributions (CPD) of each node in the graphical structure. These then became our random structures with random parameters that best described our ground truth distribution. Once we had the structure of the Bayesian network and the CPDs defined, we used forward sampling to generate random samples from the Bayesian network and created a dataset that we used for inference and learning.

For this study, we explored the below.

- We explored the following structures; tree-like, dense and sparse structures.
- We explored the Bernoulli distribution for our Conditional Probability Distributions.

3.4 Mutilating the Network

In the experimental section, we mentioned that we have a set of variables $\mathcal{X} = \{X_1, \dots, X_n\}$ and a joint distribution $P_{\mathcal{G}}(X | Y)$ that represents the relationship between them and, if we remove a variable or a subset of variables and we learn a new joint probability distribution that can be represented as $P_{\hat{\mathcal{G}}}(X | Y)$. We want to gauge how removing a variable or a subset of variables influences other nodes in the network. We will remove a random subset of the variables by starting off at 4 variables and incrementing this number randomly as we try to evaluate how much does the distribution deviate from the original distribution. When we remove variables, we will also be removing the edges that lead up to those variables that are represented as nodes in the graphical structure. We will measure this deviation by

making use of the Kullback-Leibler (KL) divergence which measures how one probability distribution diverges from a second, expected probability distribution.

3.4.1 Learning

Now that we have mutilated our graphical model, we will need to learn the new structure and parameters from the generated data. There are two main types of learning in Probabilistic Graphical Models (PGMs), namely parameter estimation and structure learning. We will first learn the graphical structure of our data from the mutilated network then proceed with estimating its parameters for our ground truth distribution. Secondly, we will learn the optimal network structure that best represents the dependencies between our variables which in this case will be ATMs. Learning in PGMs is the cornerstone for constructing models that accurately capture the underlying probabilistic relationships in the data. This allows for the creation of models that can make accurate predictions and inferences based on the available data.

3.4.1.1 Parameter Estimation

Parameter estimation refers to the process of determining the parameters of the conditional probability distributions (CPDs) associated with each node in a graph [Bishop 2006]. The probabilistic relationships between the variables are specified by these parameters and they are crucial for making predictions and performing inference. If the structure of the PGM is known, in which we do have in this case, we can use the Maximum Likelihood Estimation (MLE) and Bayesian estimation for parameter estimation. Two assumptions need to be met before we can perform parameter estimation and these are:

- That $\mathcal{D} = \xi[1], \dots, \xi[M]$ is sampled from P^* with P^* being the true or target probability distribution of variables in the model.
- Instances are independent and identically distributed (IID).

Maximum Likelihood Estimation (MLE) as stated, is one of the methods that can be used to estimate the parameters of a model based on observed data. The MLE estimate for the parameters is obtained by maximizing the likelihood function, which is the probability of observing the given data under the model. For the dataset $\mathcal{D} = x^1, x^2, \dots, x^N$ where each $x^{(i)}$ is a set of observed values for the variables in the model, the likelihood function is given by:

$$\mathcal{L}(\Theta \mid \mathcal{D} = \prod_{i=1}^N P(x^{(i)} \mid \Theta))$$

Where Θ represents the parameters of the model. The MLE estimate for Θ can be obtained by maximizing this likelihood function.

$$\hat{\Theta}_{MLE} = \arg \max_{\theta} \mathcal{L}(\Theta \mid \mathcal{D})$$

Often, the log-likelihood function is more convenient to maximize and it is expressed as follows:

$$\hat{\Theta}_{MLE} = \arg \max_{\theta} \log \mathcal{L}(\Theta \mid \mathcal{D})$$

Bayesian estimation is an alternative to MLE, it estimates the parameters by incorporating prior beliefs about the parameters. Bayesian estimation combines the likelihood of the data with a prior distribution over the parameters to compute a posterior distribution. We can encode our knowledge as a prior knowledge about θ using a probability distribution. Here we assume that the outcome is conditionally independent given the parameter θ .

$$\begin{aligned} P(x[1], \dots, x[M], \theta) &= P(x[1], \dots, x[M] \mid \theta) P(\theta) \\ &= P(\theta) \prod_{m=1}^M P(x[M] \mid \theta) \\ P(\theta \mid x[1], \dots, x[M]) &= \frac{P(x[1], \dots, x[M] \mid \theta)}{P}(\theta) P(x[1], \dots, x[M]) \end{aligned}$$

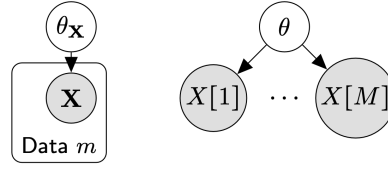


Figure 3.2: A visual representation of a dataset with θ representing the parameters.

If the prior is a Beta distribution, then the posterior is a Beta distribution. As we obtain more data, the effect of the prior diminishes. Thus the Bayesian framework allows us to trade-off a diminishing prior as more data becomes available. One common choice for a prior distribution is the Dirichlet prior. The Dirichlet distribution is a generalization of the Beta distribution. If the prior, $P(\theta)$, is Dirichlet then the posterior, $P(\theta | \mathcal{D})$, is Dirichlet. If $P(\theta)$ is $Dir(\alpha_1, \dots, \alpha_k)$, then $P(\theta | \mathcal{D})$ is $Dir(\alpha_1 + M[1], \dots, \alpha_k + M[k])$ where $M[k]$ is the number of occurrences of x^k

3.4.1.2 Structure Learning

In Bayesian networks, structure learning involves determining the optimal network structure that best represents the conditional dependencies between variables [Koller and Friedman 2009]. There are two main approaches for structure learning in Bayesian networks, namely Constraint-based methods and Score-based methods. The problem statement for structure learning can be formulated as below.

- We assume that the dataset $\mathcal{D} = \{\xi[1], \dots, \xi[M]\}$ consists of independent and identically distributed samples drawn from the true distribution $P^*(\mathcal{X})$
- $P^*(\mathcal{X})$ is induced by a Bayesian network $\mathcal{G}^*$ over $\mathcal{X}$
- To what extent do independencies manifest in $\mathcal{G}^*$ that manifest in $\mathcal{D}$

We want to find the Bayesian network structure, $\mathcal{G}$, that best represents the dependencies that manifest in $\mathcal{D}$.

Constraint-based approaches are methods that rely on statistical tests to infer conditional independence relationships between variables. These methods build an independence map (I-map) which represents the conditional independence properties of the distribution. We aim to capture the network structure that accurately represents the independencies in the domain and the basic idea is to build the best minimal I-map.

Score-based approaches are methods that assign a score to each possible network structure based on how well it fits the data. Common scoring functions include the Bayesian Information Criterion (BIC), Akaike Information Criterion (AIC), and the Bayesian Dirichlet Equivalent (BDe) score. These methods search the space of possible structures to find the one that maximizes the chosen scoring function. Examples of search algorithms include greedy search, hill-climbing, and simulated annealing.

In the case of our modelling problem for ATMs, by determining the optimal structure of a probabilistic graphical model that best fits the data, one can gain insights into the relationships and dependencies between these variables (ATM transaction volumes), which can be crucial for both understanding the underlying processes and making accurate predictions.

3.5 Evaluation

In probabilistic graphical models, inference is the process of answering probabilistic queries about the variables in a model based on observed evidence [Heckerman 1995]. Inference in PGMs can be computationally intensive, which makes it an $\mathcal{NP}$ -hard problem. However, these are results for worst case scenarios and inference with graphical models is mostly manageable for practical applications. We want to be able to compare our ground truth distribution which we generated by following the method that we discussed in the experimental setup section with the new distributions of the mutilated Bayesian networks. We will compare these distributions at the level of probabilistic queries for evaluation.

Suppose that the probability $P_B(x) = \alpha$ is the probability of x in the ground truth distribution and the probability $P_{B^\dagger}(x) = \beta$ is the probability of x in the new joint probability distribution of the mutilated network. The probability of x in this case would have changed and we can calculate the Mean Squared Error (MSE) between these two parameters by using the formula below:

$$MSE = \frac{1}{n} \sum_{i=1}^N (\alpha - \beta)^2$$

To compare the learned distribution from the ground truth distribution we will be using a well known formalism for doing this which is the KL divergence. The Kullback-Leibler (KL) divergence, is a measure used in probability theory and information theory to quantify how one probability distribution differs from a second, reference probability distribution. We will be measuring how many bits have changed from the learned distribution, and this will be done on the test data or the validation data.

Given two probability distributions P and Q defined over the same probability space, the KL divergence from Q (the approximate distribution) to P (the true distribution) is defined as:

$$D_{KL}(P||Q) = \sum_x P(x) \log \left(\frac{P(x)}{Q(x)} \right)$$

For continuous distributions, the sum is replaced by an integral:

$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} p(x) \log \left(\frac{p(x)}{q(x)} \right) dx$$

Since we are generating the ground truth distribution on our own, we are able to verify the learned model to the actual ground truth model. Unfortunately, we will not be able to do the same with the application data which is the ATM data because for that case we do not have access to the ground truth model. However, we can evaluate the model for the ATM data by using accuracy, recall and precision. There are several types of inference tasks in PGMs, we will look at variable elimination and approximate inference for performing probabilistic queries on our learned graphical model.

Figure 3.1 represents an influence structure between stochastic processes, if we reduce the graph structure by removing a node and the edges leading to P_2 we get the below influence structure.

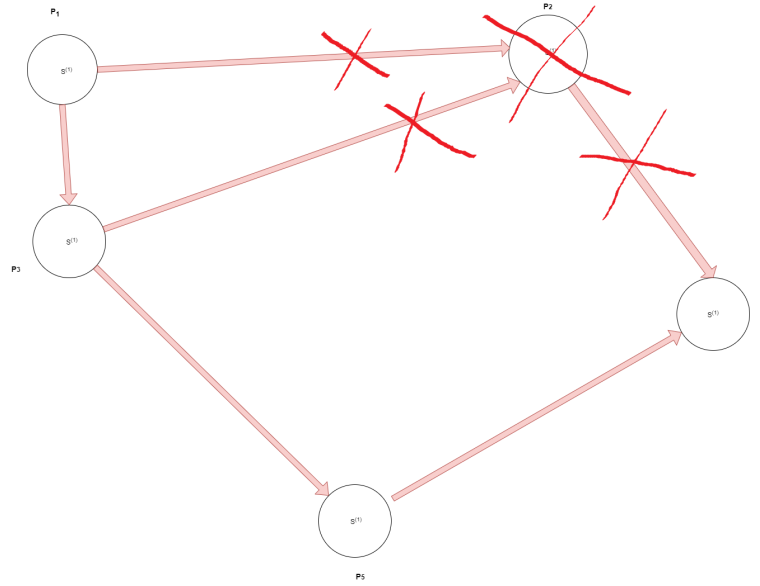


Figure 3.3: An influence structure between stochastic processes with a reduced graph structure.

If one of the parameters of a node represents a stochastic process and we remove a node as in Figure 3.3 from the network after we have learned a probabilistic graphical model, how would this influence the other nodes in the network of the learned structure and the probability density. We can also test for probabilistic inference by having an algorithm that removes nodes from the graphical model. If we let $P_{\mathcal{G}}(X | Y)$ represent the graphical model in Figure 3.1 and let $P_{\hat{\mathcal{G}}}(X | Y)$ represent the graphical model in Figure 3.3 which is the reduced graph, we know that $P_{\mathcal{G}}(X | Y) \neq P_{\hat{\mathcal{G}}}(X | Y)$. Given the reduced graph $\hat{\mathcal{G}}$, we want to test if we can still do the same inference with Y as we did with the original graph $\mathcal{G}$. In order to test for this we will utilize the 2 inference approaches that are explained below.

3.5.0.1 Variable Elimination

Exact inference refers to the computation of exact probabilities or marginal distributions of variables in the model given some evidence [Pearl 1988]. Variable elimination is an exact inference technique for eliminating variables from a joint distribution by iteratively summing out variables based on the factorization of the joint distribution. The main idea of variable elimination is to exploit the structure of the Bayesian network to simplify the computation of marginal probabilities. Below are some of the properties of the structure of a Bayesian network:

- Some sub-expressions of the joint distribution only depend on a small number of variables.
- By caching results we do not have to recompute them.
- Some important properties are:
 1. $P(A) = \sum_B P(A, B)$.
 2. $\sum_C P(A)P(B, C) = P(A) \sum_C P(B, C)$

Below is an overview of the variable elimination algorithm:

1. **Initialization** - start with the factors corresponding to the conditional probability tables (CPTs) of the variables in the network.
2. **Elimination** - for each variable X_i in the order specified do the following:

- Multiply all factors containing X_i to form a new factor F_i .
 - Sum out X_i from F_i to get new factor $\hat{F}_i$.
 - Replace the factors containing X_i with $\hat{F}_i$ in the factor list.
3. **Termination** - after eliminating all variables, the remaining factors contain the joint probability distribution of the remaining variables.
 4. **Normalization** - to obtain the marginal probabilities, normalize the factors by dividing by the sum of their values.

To reduce computational complexity of inference, we can make use of variable elimination as it eliminates redundant calculations. We can make use of variable elimination to make inference for our problem in Figure 3.3.

3.5.0.2 (Alternate) Approximate Inference

In Probabilistic Graphical Models, approximate inference refers to a collection of techniques used to approximate the posterior distribution over variables of interest when exact inference is computationally infeasible [Wainwright and Jordan 2008]. These techniques attempt to provide a close approximation to the true posterior distribution while reducing computational complexity. We will utilize two approximate inference techniques which are namely, Gibbs sampling and Markov Chain Monte Carlo (MCMC). The Gibbs sampling method attempts to “fix” the sample, by re-sampling some of the variables. It generates a sample with arbitrary values (except for the evidence), and then re-samples all variables while updating the evidence in an arbitrary order. Unlike forward sampling, the Gibbs sampling technique takes into account the downstream evidence. As we repeat the sampling process, we get closer and closer to the desired posterior:

$$P_\phi(X) = P(X \mid e)$$

The Gibbs sampling technique is an example of a class of methods called Markov Chain Monte Carlo (MCMC). MCMC is an iterative process where we move from the distribution getting closer to the desired posterior distribution. A Markov chain is defined as a graph of states over which the sampling algorithm takes a random walk. The below applies to Markov chains:

- The nodes of this graph are possible assignments of variable $x \in Val(X)$
- The edges are a transition model, $\mathcal{T}$, which specifies for each pair of states x and $\hat{x}$ the probability $\mathcal{T}(x \rightarrow \hat{x})$

3.6 Modeling the Influence of Transaction Volumes between ATMs

It turns out that the modeling problem that is formulated in section 3.1 is an important problem in real world scenarios and one of the scenarios is modeling the relationships between ATMs. Automated Teller Machines (ATMs) are ubiquitous in our society and are widely used for financial transactions. These devices play a crucial role in the financial system as they give customers the ability to withdraw money, deposit money, customer convenience and a variety of other benefits to banks and customers. These machines are often deployed at a large scale in order to serve a client base in spread out geographical locations. The size and the spread of the ATMs introduces a logistical and financial optimization problem for banks. Understanding the relationships between ATMs can be applied to various problems facing financial institutions.

Analyzing the relationship between ATMs can assist with detecting patterns of fraudulent behavior, such as coordinated attacks on multiple ATMs. Data driven decision making can enable banks to make informed decisions about whether to install a new ATM or remove an ATM at a given location based on the relationship between all the ATMs, forecast transactions and optimize the use of resources. If an ATM or a subset of ATMs in a network of ATMs were to go off due to ATM bombing, load shedding, planned maintenance etc. How would this affect the rest of the ATMs in the same region, would customers move to transact in the nearest ATM, what associations manifest in the data and how would we go about detecting vulnerabilities in the network.

Predicting the influence of the different ATMs in the network can be achieved by aggregating the features of each ATM. We will be modeling the dependencies and relationships between different ATMs and their transaction volumes by using a probabilistic graphical model. The variables of study will be mainly the ATM transaction volumes, each node in the Bayesian network will represent a variable. The nodes are the variables and they are used to represent ATMs. In our case, each node will represent the transaction volumes of individual ATMs. The probability distribution of each node which is used to represent ATMs is going to be transaction volumes.

There will be local probability distributions inside of each of these nodes that models the ATM transaction volumes. The Conditional Probability Distribution for the variable is a function of the variable itself and the parents. This is modeled as a local probability distribution, it can be modeled using a tree structure, tree CPD, tabular CPD and a Gaussian CPD. In our case where we are studying transaction volumes which are continuous variables, we have chosen Gaussian CPDs. The dependencies between the nodes will be represented by edges, for an example the transaction volumes at an ATM might depend on the transaction volumes at nearby ATMs, the time of day, or external factors like holidays.

Historical ATM transaction data will then be used to learn the Conditional Probability Distributions and the structure of the Bayesian network. The structure of the Bayesian network will be determined by using structure learning algorithms. Structure learning is the process of determining the graph structure that best represents the dependencies among the variables in the model. Parameter learning will be used to estimate the CPDs for each node given the structure. To analyze the outcomes of the modeling process we will utilize variable elimination for exact inference. Once we have a structure that represents the dependencies between the nodes and their CPDs, we can mutilate the network by removing a node or a subset of nodes and the edges leading up to these nodes.

The method of removing nodes means that we specifying the nodes that we are dropping on the graphical structure which would modify the structure of the model. Parameter estimation will then be used to learn the new parameters of the modified structure so that exact inference can be used to see how the removal of nodes affects the outcomes. Model evaluation will follow the same steps as outlined in section 3.4 and to measure the deviation between the original structure of the Bayesian network and the new structure where nodes are removed we will use Kullback-Leibler (KL) divergence which measures

how one probability distribution diverges from a second, expected probability distribution.

The figure below, Figure 3.4 is a visual representation of ATMs for a South African bank in the Johannesburg region, the yellow circles with white stars in the centre represent individual ATMs. If the ATMs marked with a red X were to go off due to ATM bombing, load shedding, planned maintenance etc. How would this affect the rest of the ATMs in the same region, how would we go about detecting vulnerabilities in the network.

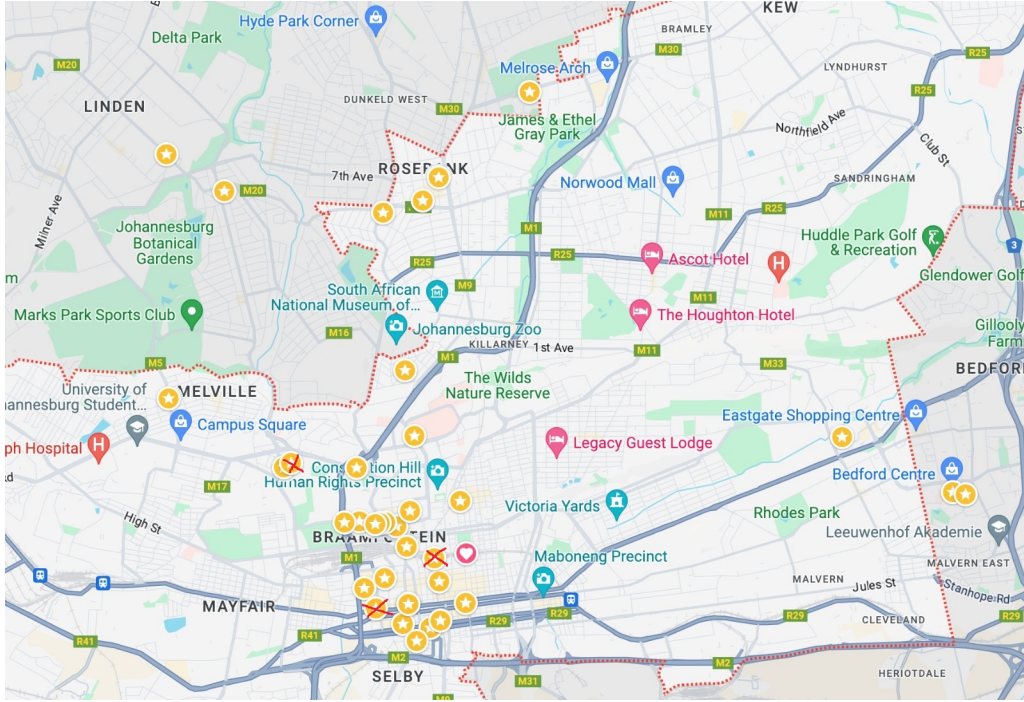


Figure 3.4: A distribution of ATM networks in the Johannesburg region for a South African bank. The yellow circles with white stars in the center indicate an individual ATM, the red crosses on these ATMs indicate a hypothetical event of removing an ATM from a network.

3.6.1 Data Preprocessing

We obtained a dataset for 156 ATMs that are situated in Johannesburg from a South African bank spanning a period of 4 years, from 2021 to 2024 and these were daily transaction volumes. Confidential identifiers and sensitive bank data were masked. The following features were identified for ATMs for the purpose of our study: ATM ID (unique identifier), Longitude and Latitude, and Daily Transaction Volumes.

3.6.2 Data and Ethical Implications

The data was made available by the bank and the necessary sign offs were obtained. Ethics clearance has been obtained from the University and all supporting documentation for the application has been stored appropriately. The following conditions were applied to the data; the bank cannot be named as the research site, the report may not contain customer or landlord information, and lastly, the names of the ATMs must be amended so that the machine learning outputs do not link directly back to the bank. Synthetic data was also used as part of the modeling process to answer the research questions. We adhered to all institutional data-handling and privacy guidelines to ensure confidentiality.

3.6.3 Practical Implementation and Challenges

To bridge the gap between our proof-of-concept experiments on synthetic Bayesian networks and their deployment in operational ATM networks, we must consider several practical implementation challenges and corresponding mitigation strategies. First, data integration and preprocessing present significant hurdles. Transaction logs are often scattered across heterogeneous systems, relational SQL databases, and NoSQL stores—requiring a unified ETL pipeline that can reliably extract daily withdrawal and deposit records, transform them into a consistent schema, and load them into a centralized analytics platform. In doing so, one must navigate strict banking IT governance rules, including asset tagging and role-based access controls. Moreover, real-world ATM feeds frequently exhibit missing or noisy records due to outages or network latency; these anomalies must be flagged by rule-based filters (for example, volume spikes exceeding historical maxima) and corrected through localized time-series interpolation or median-based imputation to ensure the downstream inference remains robust.

Scaling the core Bayesian structure-learning algorithms from tens or hundreds of nodes to thousands of ATMs introduces computational challenges. Full hill-climb or tree-search methods become prohibitive at that scale. A practical remedy is to partition the network by geography or usage clusters and learn sub-network structures independently before integrating them via meta-learning techniques. Incremental, online learning algorithms can also update existing DAGs as new data arrives, dramatically reducing computational load compared to full retraining. Furthermore, parallelizing structure-learning tasks across a compute cluster—using MPI or Spark—allows each machine to process a subset of nodes, thus distributing both memory and CPU requirements.

Operational complexity extends beyond initial model training to ongoing maintenance and versioning. As ATMs are added, relocated, or decommissioned, portions of the learned DAG must be updated; employing a version control system (e.g., Git) for model artifacts and a CI/CD pipeline ensures that each structural adjustment is automatically tested on historical data before deployment. To guard against behavioral drift—such as shifts in customer usage patterns during holidays or external events—statistical monitors like the Population Stability Index can trigger retraining workflows when significant distributional changes are detected.

Latency and deployment constraints must also be addressed. While batch inference may suffice for periodic network health reports, real-time risk scoring (identifying which ATMs are critical at any moment) demands sub-second query performance. Approximate inference techniques, such as loopy belief propagation or precompiled conditional probability tables, can provide rapid responses. Given compliance and security concerns, many banks rely on on-premise hardware without GPUs or TPUs; containerized solutions (Docker/Kubernetes) ensure that inference engines can scale flexibly across available CPU resources while maintaining auditability.

Finally, robust validation and governance frameworks are essential. Before rolling out any model changes, predictions should be benchmarked on a hold-out dataset of recent transactions and compared against observed service disruptions—such as customer rerouting rates—to calibrate acceptable thresholds. Explainability tools, including sensitivity analysis or Shapley-value breakdowns on the learned network, make it possible to justify why certain ATMs are flagged as critical nodes, satisfying audit and regulatory requirements. As fallback mechanisms, simple heuristics—such as ranking ATMs by historical volume—can serve when model outputs are delayed or unavailable, ensuring uninterrupted operational decision-making. By attending to these data, computational, and governance challenges, our Bayesian-network framework can transition from controlled experiments to a scalable, real-world tool for enhancing ATM network resilience.

3.7 Conclusion

This chapter detailed the methodology used to investigate the impact of node removal on Bayesian networks representing ATM transaction networks. The chapter begins by formulating the problem mathematically, defining the ATM network as a directed acyclic graph with conditional probability distributions assigned to each node. In doing so, it establishes the fundamental framework within which network disruptions are analyzed.

Subsequently, the chapter describes the experimental setup and the procedures for data collection and preprocessing. The design of the experiments is explained in detail, including the development of structure learning algorithms, the implementation of variable elimination for exact inference, and the application of approximate inference methods. The use of both exact and approximate inference techniques is discussed in the context of recovering network structure and quantifying changes using metrics such as KL divergence.

The methodology presented provides a systematic approach for addressing the research questions. It explains how the influence between variables is measured before and after node removal, ensuring that the changes in the joint probability distribution are captured rigorously. Moreover, the chapter outlines the specific procedures for learning and evaluating the network structure, thereby establishing the basis for the experimental analysis described in subsequent chapters.

Overall, the methodology detailed in this chapter serves as a robust foundation for the investigation. It is structured to allow for replicability and clarity in the recovery of network influence after disruption. The approach combines mathematical formulation with practical steps for data preprocessing, model training, and evaluation, making it possible to assess the resilience of ATM networks under varying conditions.

Chapter 4

Experimental Results and Discussion

In this chapter we present and analyze the findings related to Bayesian network recovery with varying structural configurations; tree-like, sparse, and dense structures. We generate random Ground Truth Distributions for each structural configuration with varying complexity and random parameters. We generate varying sample sizes and use structure learning to recover the models from these samples. Mainly the Hill Climb Search algorithm with the Bayesian Information Criterion are used for the structure learning with the Bayesian estimation that uses the Dirichlet prior for estimating the parameters of the recovered models.

We make use of the Kullback-Leibler (KL) divergence to compare the recovered models to the ground truth distributions. We then attempt to remove varying percentages of nodes from the recovered models in order to address the goal of understanding how different network structures respond to mutilation and the effects of increasing the training data on network recovery accuracy. Lastly, we use real ATM data to demonstrate the applications of the findings from the synthetic data experiments that were conducted.

4.1 Research Questions 1 and 2: What is the impact on the KL divergence when nodes are removed from a recovered Bayesian network? & What are the Implications of Varying Structural Complexity on the Recovery of Bayesian Networks and their KL Divergence?

In the methodology chapter we described ground truth distributions (GTDs), these will be randomly generated by randomizing the edge orientations and assigning random Bernoulli Conditional Probability Distributions (CPDs) to each node. Multiple (10) Directed Acyclic Graphs (DAGs) are generated for different sample sizes in order to add robustness across different structures. The structural configurations that we used for GTDs being; tree-like, sparse and dense structures. We used forward sampling to generate training data from the GTDs for varying sample sizes.

A 80/20 percent split was done on each sample size, 80% of the data was used for training or performing model recovery with structure learning and 20% of the data was used for testing when calculating the KL divergence of the recovered structures to the original structures. The KL divergence is computed to measure the difference between the true model's Conditional Probability Distributions and the recovered model's Conditional Probability Distributions. The mean KL divergence is computed for each sample size by taking the KL divergence values for each trial and summing up all these values and dividing by 10.

The standard error of the mean is used for the error bars to quantify the precision of the mean as an estimate of the same population, the SEM is calculated simply by taking the standard deviation and dividing it by the square root of the sample size. Validation through the KL divergence and the training and testing split uses a robust approach that calculates the KL divergence only on the testing data. This reduces over-fitting and ensures that the model’s generalization capability is being measured accurately.

All resulting models that are recovered with structure learning have their parameters estimated with Bayesian estimation using Dirichlet priors. Dirichlet priors incorporate prior beliefs, this provides robust learning when needing to avoid over-fitting in Bayesian network models.

4.1.1 Tree-like Structures

A tree-like structure in graphical models, especially in Bayesian networks, is a Directed Acyclic Graph (DAG) where nodes are connected in a hierarchical pattern with no cycles. Each node (except the root) has exactly one parent, and there is only one unique path between any two nodes. In terms of connectivity, nodes are arranged such that there is a clear parent-child relationship, creating a top-down or bottom-up flow of influence. There is only one path between any 2 nodes, which simplifies the structure and makes the task of inference more efficient. If one were to remove a node from a tree-like structural configuration, the local impact would be on the specific branch or sub-tree rather than the entire network. Tree-like structures are interpretable especially for causal relationships but they may not capture complex interactions or dependencies between nodes outside of the hierarchical structure.

Figure 4.1 visualizes ten directed acyclic graphs (DAGs) generated to represent tree-like ground truth distributions (GTDs). These graphs have a hierarchical structure where each node has at most one parent, and there are no cycles or multiple paths between nodes. Edge orientations were randomized to introduce variability across trials, and conditional probability distributions (CPDs) for each node were assigned using the Bernoulli distribution, reflecting discrete probabilistic dependencies. Each DAG represents a different trial to evaluate the recovery performance of structure learning algorithms under tree-like configurations.

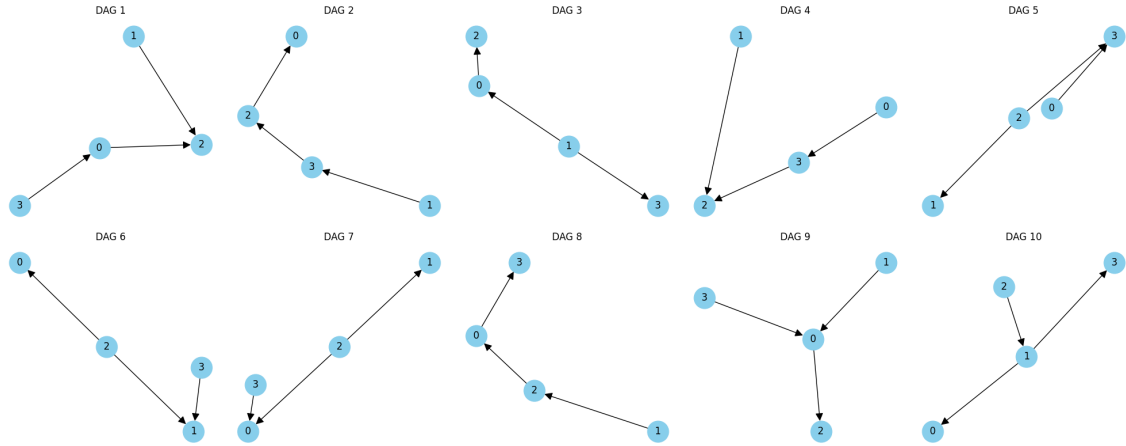


Figure 4.1: 10 randomly generated tree-like DAGs with random edge orientations and random Bernoulli CPDs assigned to each node.

Figure 4.2 displays a snapshot of the first ten samples generated from one of the tree-like ground truth distribution (GTD) directed acyclic graphs (DAGs). These samples were created using forward sampling, a technique where data points are generated based on the probabilistic dependencies defined

in the CPDs of the DAG. Each row in the snapshot represents a unique observation, with the columns corresponding to the nodes in the GTD.

Top 10 Samples:

	0	58	1	30	2	74	133	3	71	4	...	24	90	95	70
0	0	0	0	0	1	1	1	0	1	1	1	...	1	1	1
1	1	0	1	0	0	1	0	1	0	0	...	1	1	1	0
2	1	1	0	1	1	0	0	1	0	1	...	1	0	1	1
3	1	0	0	1	1	0	1	1	1	0	...	1	1	1	0
4	1	0	0	1	1	0	1	0	1	1	...	0	1	0	1
5	0	0	0	1	0	1	0	1	0	0	...	1	1	0	1
6	1	1	0	0	0	0	0	1	0	1	...	0	0	0	1
7	1	1	1	0	0	0	0	1	1	0	...	1	0	1	1
8	0	1	0	1	1	0	0	1	1	1	...	1	1	0	1
9	1	0	0	1	0	0	1	0	1	0	...	1	1	1	1

Figure 4.2: A snapshot of the top 10 samples that are generated from one of the GTD DAGs.

Figure 4.3 compares the performance of the Tree Search and Maximum Weighted Spanning Tree (MWST) algorithms in recovering tree-like GTDs with 150 nodes. The x-axis represents the size of the training dataset, and the y-axis shows the KL divergence, measuring the deviation of the recovered model from the ground truth. Smaller KL divergence values indicate better recovery of the original structure.

The MWST consistently achieves lower KL divergence than the Tree Search algorithm, especially with smaller training datasets. Both algorithms show convergence as the training data increases, but MWST demonstrates superior performance in recovering large tree-like structures. This suggests that MWST's ability to maximize dependencies between connected nodes makes it more effective for tree-like configurations with a large number of nodes.

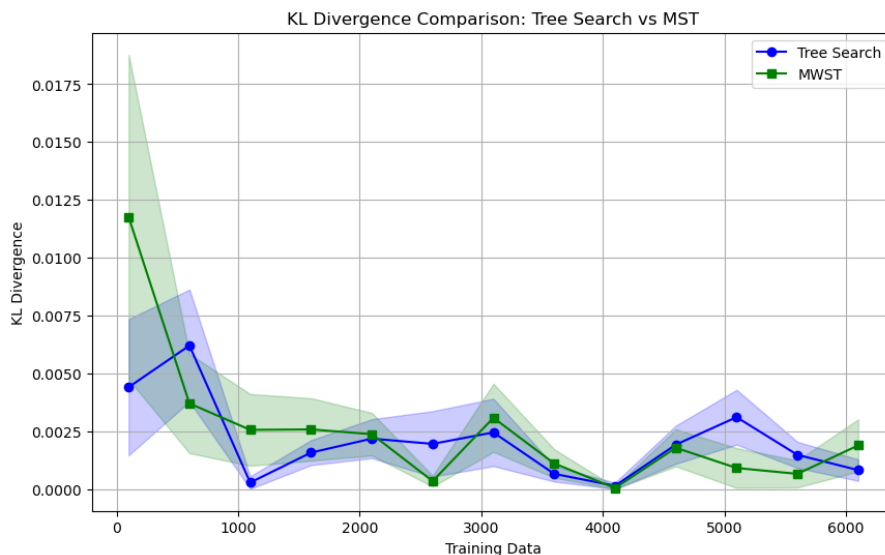


Figure 4.3: A plot of the training data versus the KL divergence for GTDs with 150 nodes comparing the Tree Search and MWST algorithms.

Figure 4.4 illustrates the performance of the MWST algorithm in recovering tree-like GTDs with 10 nodes. The x-axis denotes the size of the training dataset, while the y-axis represents the KL divergence.

As the training data increases, the KL divergence decreases significantly, indicating improved recovery of the ground truth network. The error bars, representing the standard error of the mean (SEM), also shrink with more training data, showing consistent recovery performance. This result highlights the critical role of sufficient training data in ensuring accurate recovery, even for relatively small networks.

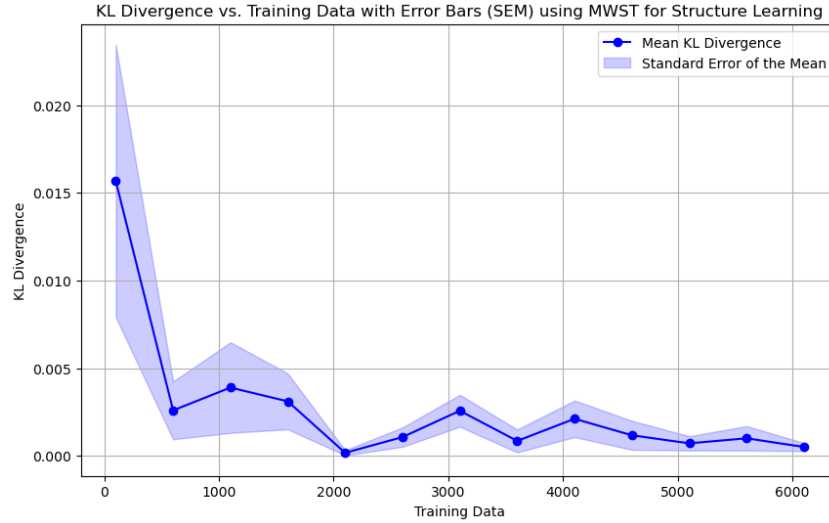


Figure 4.4: A plot of the training data versus the KL divergence for GTDs with 10 nodes using MWST for structure learning.

Figure 4.5 depicts the effects of node removal (mutilation) on the recovery performance of tree-like GTDs with 10 nodes. Results are shown for networks with 0%, 20%, and 40% node removal. Node removal significantly increases KL divergence, with 40% mutilation causing the largest deviations. The results suggest that even minor disruptions in tree-like networks can lead to considerable deviations in structure recovery, emphasizing the importance of protecting key nodes in such configurations.

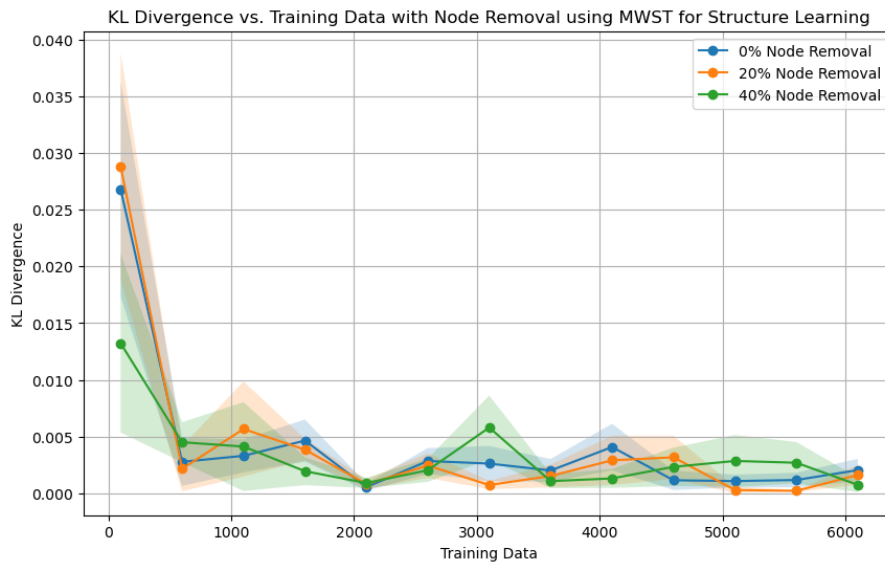


Figure 4.5: A plot of the training data versus the KL divergence for GTDs with 10 nodes using MWST for structure learning with mutilation.

Figure 4.6 evaluates the recovery performance of MWST for tree-like GTDs with 50 nodes. Without node removal, the KL divergence decreases steadily with more training data, mirroring the trends seen in smaller networks.

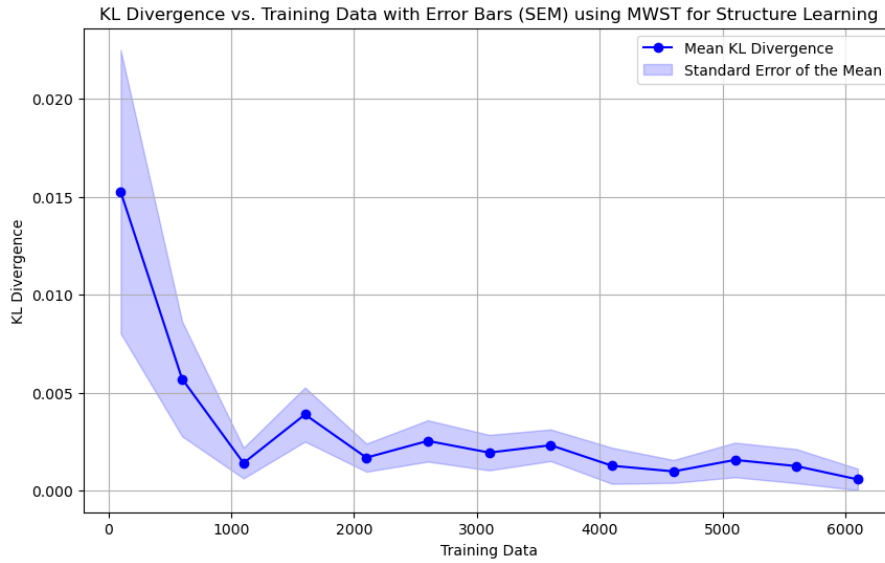


Figure 4.6: A plot of the training data versus the KL divergence for GTDs with 50 nodes using MWST for structure learning.

Figure 4.7 includes 20% and 40% mutilation scenarios. Mutilation for Figure 4.7, the KL divergence increases significantly, particularly with 40% node removal. The larger node count amplifies the sensitivity to disruptions, as tree-like networks rely heavily on hierarchical paths. These results highlight the importance of redundancy or alternative pathways in larger networks to mitigate the impact of node removal.

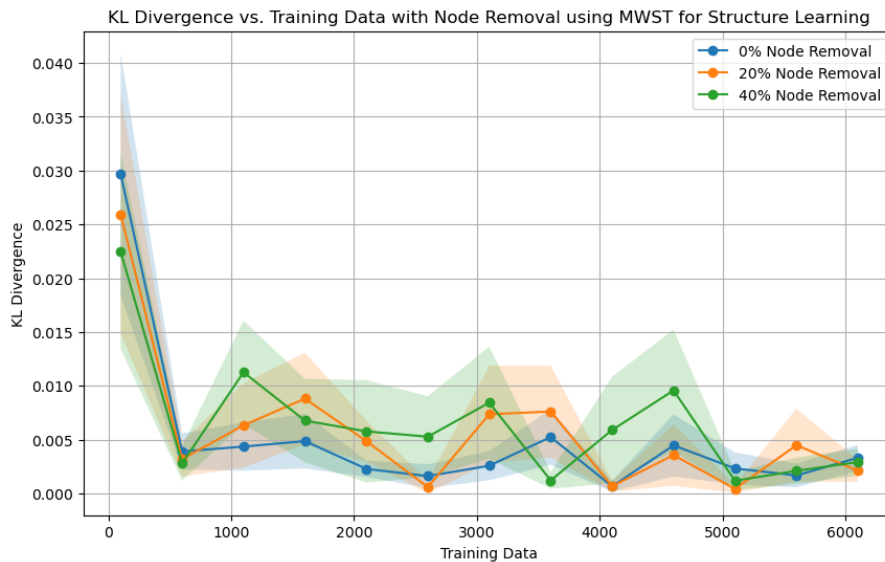


Figure 4.7: A plot of the training data versus the KL divergence for GTDs with 50 nodes using MWST for structure learning with mutilation.

Figure 4.8 shows the KL divergence as a function of training data size, providing insights into the recovery performance of tree-like networks as data volume increases. The figure demonstrates a rapid decrease in KL divergence as the training data size increases. This indicates that tree-like structures are particularly effective in stabilizing their recovery with smaller datasets.

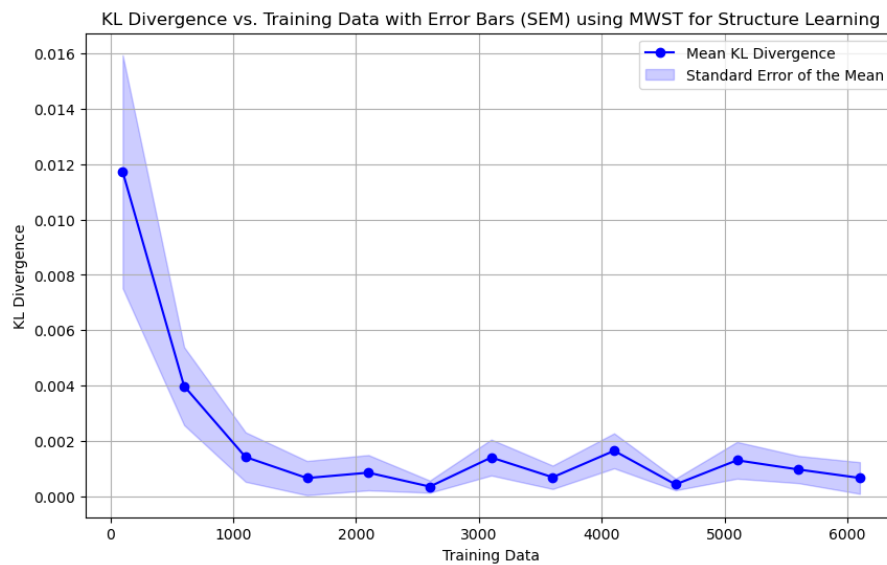


Figure 4.8: A plot of the training data versus the KL divergence for GTDs with 100 nodes using MWST for structure learning.

Figure 4.9 includes 20% and 40% mutilation scenarios. Mutilation for Figure 4.9, the KL divergence increases significantly, particularly with 40% node removal. As expected, larger datasets lead to improved recovery and lower KL divergence for networks without node removal. In the mutilated networks, the impact of 40% node removal is particularly pronounced, resulting in significant deviations. Larger networks with more nodes suffer greater cascading effects from mutilation, further underscoring the fragility of tree-like structures when key nodes are removed.

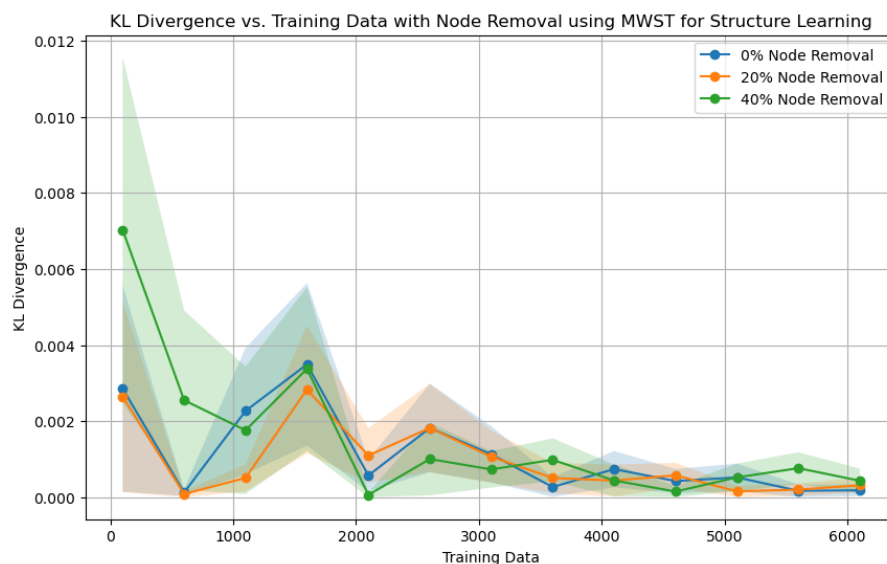


Figure 4.9: A plot of the training data versus the KL divergence for GTDs with 100 nodes using MWST for structure learning with mutilation.

Figure 4.10 depicts the KL divergence as training data size increases for a 150 node tree-like structure. The KL divergence decreases with increasing training data, following a similar trend to smaller networks. However, the stabilization point occurs at higher training dataset sizes compared to the 100 node network, reflecting the added complexity of the larger structure.

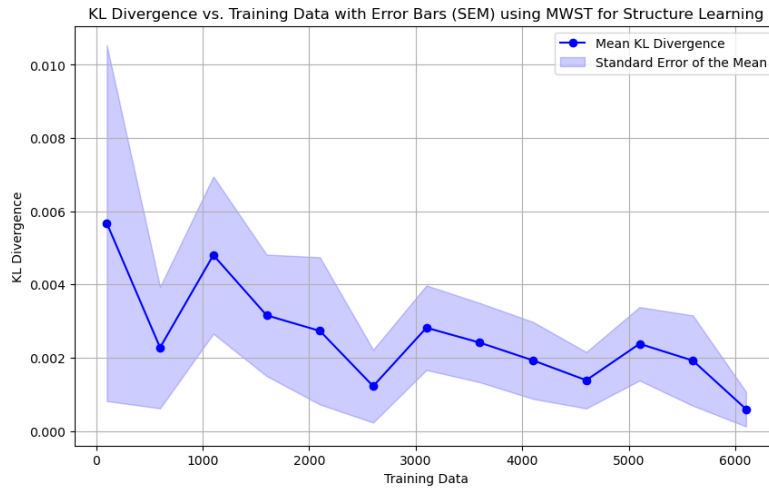


Figure 4.10: A plot of the training data versus the KL divergence for GTDs with 150 nodes using MWST for structure learning.

Figure 4.11 illustrates the KL divergence between the ground truth and the recovered model as a function of node removal. The figure reveals a steep increase in KL divergence with incremental node removal. The simplicity of the tree-like structure, while advantageous for recovery with smaller datasets, makes it particularly sensitive to the loss of nodes. The KL divergence rises sharply even at lower percentages of node removal, indicating the criticality of preserving central nodes in maintaining the network's integrity. The findings emphasize the vulnerability of tree-like structures to node removal. While these structures excel in efficiency and interpretability, their resilience heavily depends on the preservation of critical nodes. For ATM networks, this suggests that tree-like models are best suited for applications where critical nodes can be safeguarded, ensuring continuity of the modeled relationships. Robust backup mechanisms or redundancy at key points may mitigate the impact of node removal in such setups

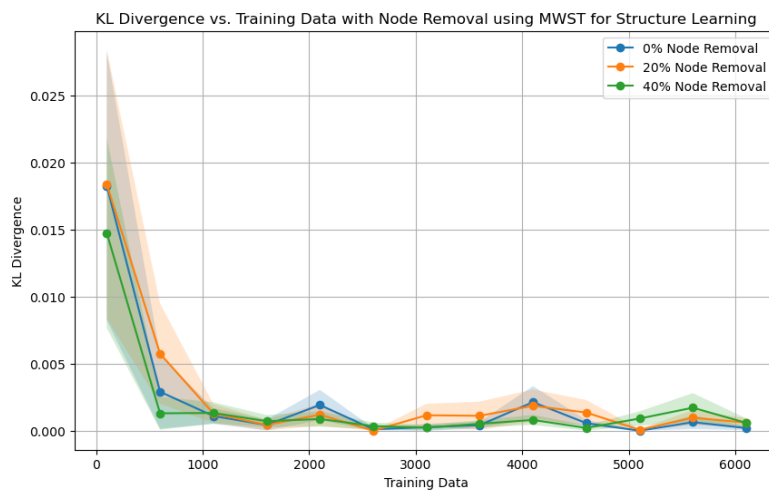


Figure 4.11: A plot of the training data versus the KL divergence for GTDs with 10 nodes using MWST for structure learning with mutilation.

4.1.1.1 Discussion of observations and implications

Figure 4.1 depicts 10 tree-like randomly generated DAGs with random Bernoulli CPDs and edge orientations and each DAG is treated as a different trial for each sample size and these are again randomly generated with different parameters and edge orientations for each sample size. The DAGs are sampled and they produce samples that are depicted in Figure 4.2. Structure learning is performed for each sample size, then the recovered models have their CPDs estimated with Bayesian estimation using Dirichlet priors. The KL divergence is computed for all 10 trials, and then we take the mean KL divergence and we plot it with the error bars of that sample size given by the Standard Error of the Mean (SEM).

Figure 4.3 compares the performance of the Maximum Weighted Spanning Tree and the Tree Search algorithms on recovering models from samples that are generated by tree-like ground truth distributions. The MWST finds the tree structure that maximizes the total dependency between connected variables and this ensures that the learned structure reflects the most significant probabilistic relationships. On the other hand, the tree search algorithm uses MWST as part of searching for trees but adds an additional step to orient edges to create a valid Bayesian network (DAG). Figure 4.4 is a plot of the training data versus the KL divergence where each GTD has 10 nodes. When the training data starts to increase, we observe that the KL divergence starts to converge around zero and this suggests that the more training data we have, the better the recovery of the original model with structure learning. The SEM error bar sizes also decrease with more training data.

To simulate the removal of ATMs, we randomly remove 20% and 40% of the nodes and the edges leading to or away from these nodes and this is done on the recovered models. We use the testing data to compute the KL divergence to the GTDs. Figure 4.5 depicts the effects of mutilating the recovered models starting with 10 nodes on the GTDs. We then modify the complexity of the GTDs by increasing the number of nodes starting from 10, 50, 100 until 150. Figure 4.6 is the recovered model when the GTDs have 50 nodes, while Figure 4.7 are the recovered models that are mutilated. Figure 4.8 and 4.9 are the results when we have set the number of nodes to 100. Lastly, Figure 4.10 and Figure 4.11 are for GTDs with 150 nodes.

Tree-like structures are highly sensitive to node removal due to their hierarchical design, where the removal of a single node can disrupt the entire sub-tree. Larger networks with more nodes show greater cascading effects from mutilation, resulting in higher KL divergence. The MWST algorithm consistently demonstrates strong recovery performance for tree-like structures, particularly with sufficient training data. Increasing the size of the training dataset improves recovery accuracy and reduces variability across trials, emphasizing the importance of data sufficiency in structure learning.

Tree-like networks require careful identification and protection of critical nodes to minimize disruptions caused by node removal. Ensuring access to large and representative datasets is crucial for accurate recovery of hierarchical structures. The MWST algorithm's ability to leverage dependency relationships makes it a suitable choice for recovering tree-like Bayesian networks, particularly for applications where simplicity and efficiency are priorities. In contexts such as ATM networks, tree-like structures may be effective for modeling dependencies but must be complemented by redundancy strategies to ensure robustness against outages or disruptions.

4.1.2 Sparse Structures

A sparse structure in graphical models refers to a graph with relatively few edges compared to the number of nodes. This translates to a node being connected to only a small subset of other nodes, leading to minimal connectivity between distant nodes and fewer dependencies. In terms of connectivity, nodes are not heavily interconnected and information flow is restricted to smaller, local sub-graphs, with fewer

pathways between any two nodes. If one were to remove a node from a sparse structural configuration, this may not critically affect the entire network, especially if connections are localized.

Figure 4.12 displays 10 sparse Directed Acyclic Graphs (DAGs) generated to serve as ground truth distributions (GTDs) for evaluating structure learning algorithms. Sparse structures are characterized by a low number of connections per node, ensuring minimal dependencies between variables. The edge orientations were randomized to introduce structural variability, and Bernoulli distributions were used to assign Conditional Probability Distributions (CPDs) to each node, reflecting discrete probabilistic dependencies. These sparse DAGs provide a controlled yet diverse set of experimental conditions for studying the recovery performance of structure learning methods under minimal connectivity scenarios.

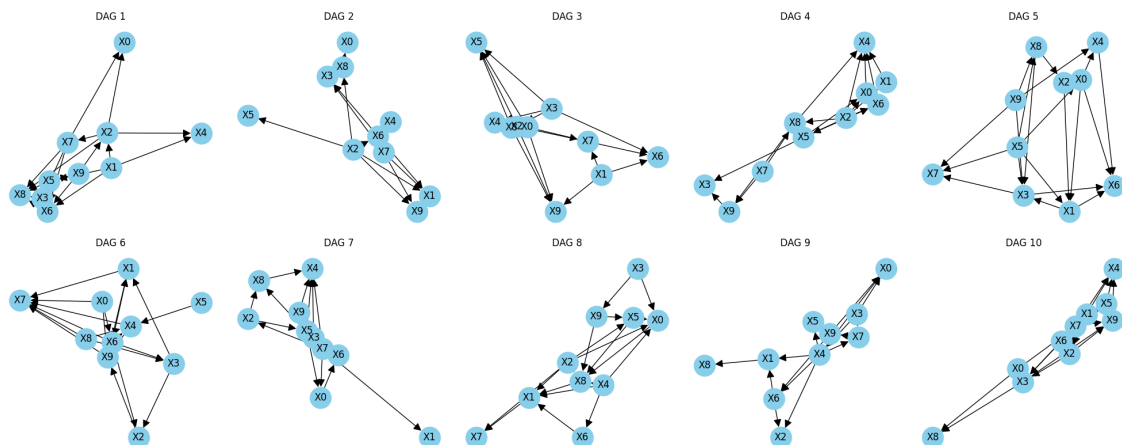


Figure 4.12: 10 randomly generated sparse DAGs with random edge orientations and random Bernoulli CPDs assigned to each node.

Figure 4.13 provides a snapshot of the first 10 data samples generated from one of the sparse Ground Truth Distribution (GTD) Directed Acyclic Graphs (DAGs). Each row represents a single observation, while each column corresponds to a variable (node) in the DAG. The samples were generated using forward sampling, which produces data consistent with the probabilistic dependencies defined by the DAG's structure and the Bernoulli-distributed Conditional Probability Distributions (CPDs).

Top 10 samples of the first sample size:

	X1	X10	X2	X6	X3	X7	X5	X4	X8	X9
0	0	0	1	1	1	0	1	0	0	1
1	0	0	1	0	1	0	1	0	0	1
2	0	0	0	1	0	0	1	0	0	1
3	0	0	1	1	0	0	1	0	0	1
4	0	0	1	1	0	1	1	0	0	1
5	1	1	1	1	0	0	1	0	0	0
6	1	0	0	1	0	0	1	0	0	1
7	0	0	1	0	1	0	1	0	0	1
8	0	0	0	1	1	0	1	0	0	1
9	1	1	1	1	0	0	1	0	0	1

Figure 4.13: A snapshot of the top 10 samples that are generated from one of the GTD DAGs.

Figure 4.14 illustrates the performance of three scoring metrics and these are; the Bayesian Information Criterion (BIC), Kishino-Hasegawa Information Criterion (KIC), and the Bayesian Dirichlet Score

(BDScore) in recovering sparse Ground Truth Distributions (GTDs) with 10 nodes. The x-axis represents the size of the training dataset, while the y-axis measures the KL divergence between the recovered network and the ground truth. Lower KL divergence values indicate better recovery of the original network. The experiments were conducted using the Hill Climb Search algorithm, a score-based method for learning the structure of Bayesian networks. This comparison highlights how the choice of scoring metric influences the accuracy of the recovered structure. The results underscore the importance of selecting an appropriate scoring method for structure learning. BIC emerges as the most reliable metric for sparse configurations, particularly when training data is limited. This finding is crucial for real-world applications where data availability may vary, as it highlights BIC's robustness in balancing accuracy and complexity. By comparing scoring methods, this figure emphasizes the role of algorithmic choices in recovering Bayesian networks, providing actionable insights for practitioners designing systems with sparse probabilistic dependencies.

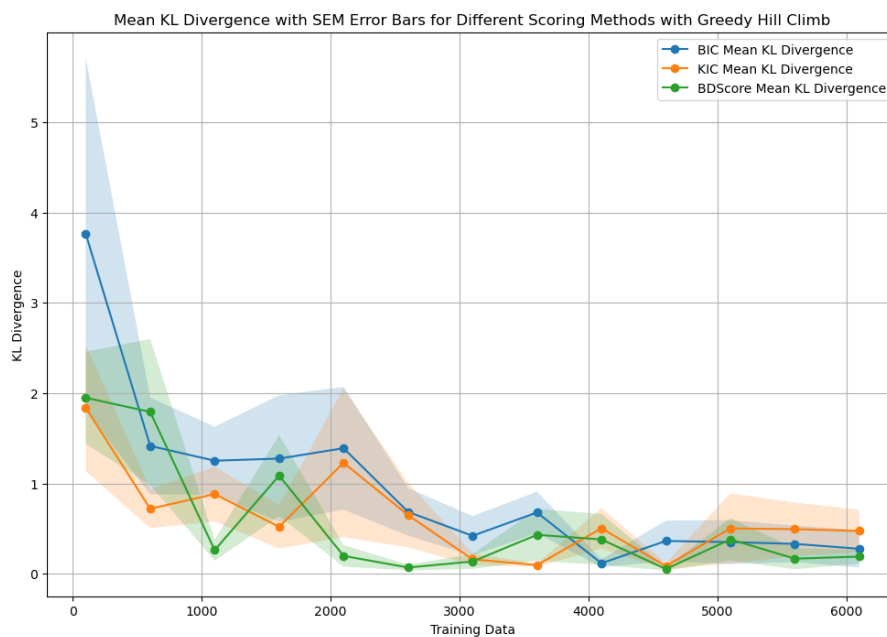


Figure 4.14: A plot of the training data versus the KL divergence for GTDs with 10 nodes comparing the BIC, KIC and BDScore for scoring while using the Hill Climb Search algorithm for structure learning.

Figure 4.15 shows the performance of the Hill Climb Search algorithm in recovering sparse Ground Truth Distributions (GTDs) with 10 nodes, using the Bayesian Information Criterion (BIC) as the scoring method. The x-axis represents the size of the training dataset, while the y-axis indicates the KL divergence between the recovered network and the ground truth. Lower KL divergence values reflect better recovery accuracy. The experiment evaluates how the amount of training data influences the algorithm's ability to reconstruct the sparse network, providing insights into the data sufficiency requirements for effective structure learning. The results highlight the critical role of data size in the performance of structure learning algorithms. Sparse networks, due to their limited dependencies, are more recoverable with smaller datasets, but data sufficiency is still essential to achieve high accuracy. The figure also reaffirms BIC's effectiveness in sparse configurations, where its penalty for model complexity ensures that extraneous dependencies are not introduced during recovery. These insights can guide the design of probabilistic models in applications where data is limited but accuracy is paramount, such as sensor networks or lightweight decision systems. This analysis lays the groundwork for understanding how scoring metrics and training data interplay to affect the recoverability of sparse Bayesian networks.

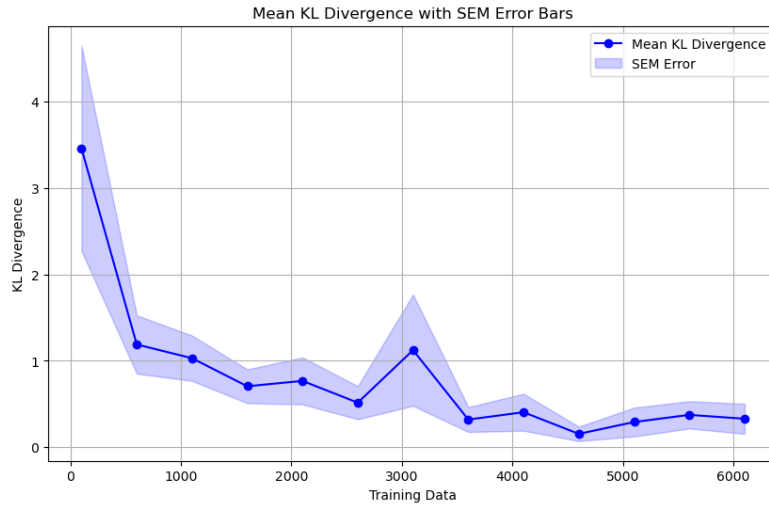


Figure 4.15: A plot of the training data versus the KL divergence for GTDs with 10 nodes using Hill Climb Search with BIC for scoring.

This Figure 4.16 illustrates the impact of node removal (mutilation) on the recovery performance of sparse Ground Truth Distributions (GTDs) with 10 nodes. The Hill Climb Search algorithm is employed for structure learning, using the Bayesian Information Criterion (BIC) as the scoring method. The x-axis represents the size of the training dataset, and the y-axis displays the KL divergence, measuring the deviation of the recovered network from the ground truth. Figure 4.16 underscores the vulnerability of sparse networks to node removal. Since sparse networks have fewer connections, the removal of even a small number of nodes disrupts dependencies significantly, leading to increased KL divergence. The partial recovery observed with moderate mutilation suggests that these networks benefit from redundancy or additional data to compensate for structural loss.

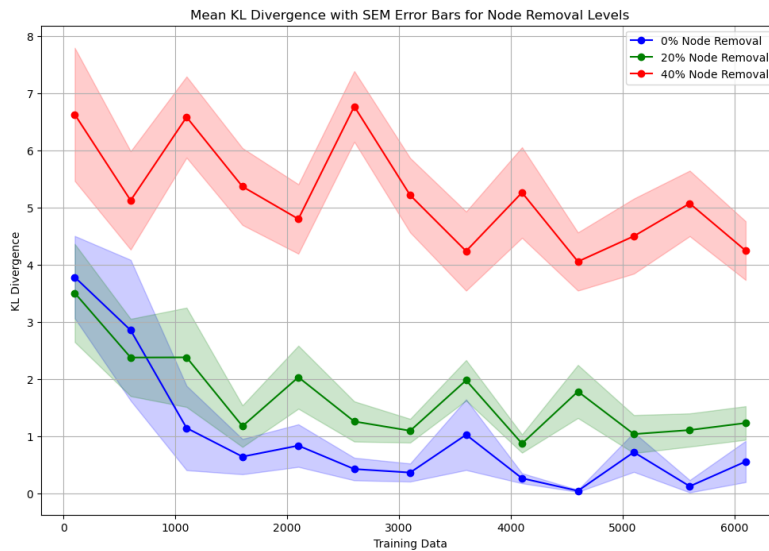


Figure 4.16: A plot of the training data versus the KL divergence for GTDs with 10 nodes using Hill Climb Search with BIC for structure learning with mutilation.

Figure 4.17 presents the performance of the Hill Climb Search algorithm in recovering sparse Ground Truth Distributions (GTDs) with 15 nodes, using Bayesian Information Criterion (BIC) as the scoring metric. The x-axis represents the size of the training dataset, while the y-axis shows the KL divergence, which measures the difference between the recovered network and the original ground truth. The experiment evaluates how the recovery accuracy evolves with increasing training data, offering insights into the scalability of the algorithm as the network size increases.

Figure 4.17 demonstrates that sparse networks with 15 nodes require more data for accurate recovery compared to smaller networks (e.g., 10 nodes). This result reflects the increased complexity introduced by a larger network, where more dependencies must be learned from the data. These findings highlight the importance of ensuring adequate data availability for structure learning in moderately sized networks. Additionally, the consistent performance of BIC reaffirms its suitability for sparse configurations, where its ability to penalize model complexity prevents overfitting.

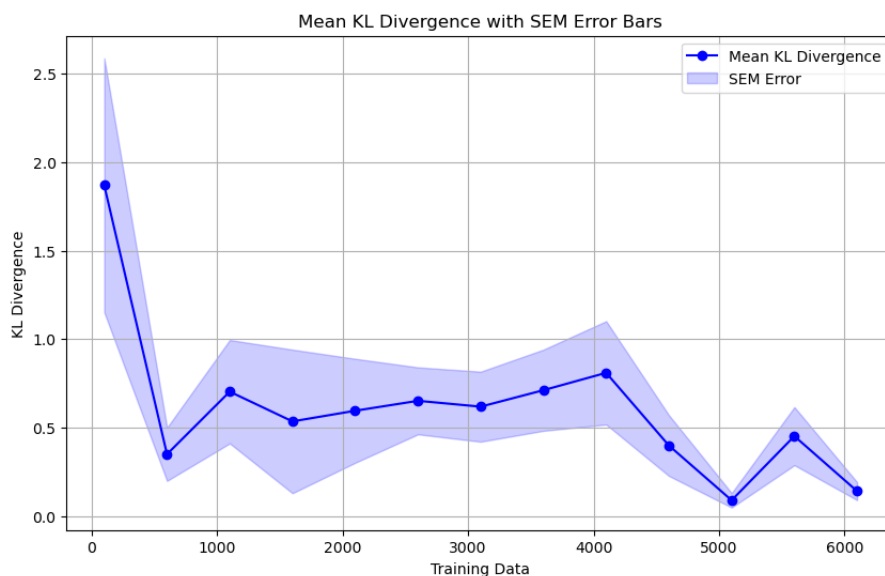


Figure 4.17: A plot of the training data versus the KL divergence for GTDs with 15 nodes using Hill Climb Search with BIC for scoring.

Figure 4.18 introduces node removal (mutilation) scenarios. The Hill Climb Search algorithm, using BIC as the scoring method, is applied to recover GTDs with 15 nodes under the following conditions: 0% node removal where no nodes are removed, representing the ideal recovery scenario. 20% node removal where moderate structural damage is introduced. Lastly, 40% node removal where severe structural damage is applied. This figure highlights the challenges posed by node removal in moderately sized networks. Sparse networks with 15 nodes are particularly sensitive to disruptions, as each removed node represents a larger proportion of the total network. Additionally, the importance of sufficient training data is underscored, as larger datasets can partially compensate for moderate structural damage. However, the results also show that data alone may not be enough to fully recover networks with severe mutilation, highlighting the need for alternative strategies in such scenarios.

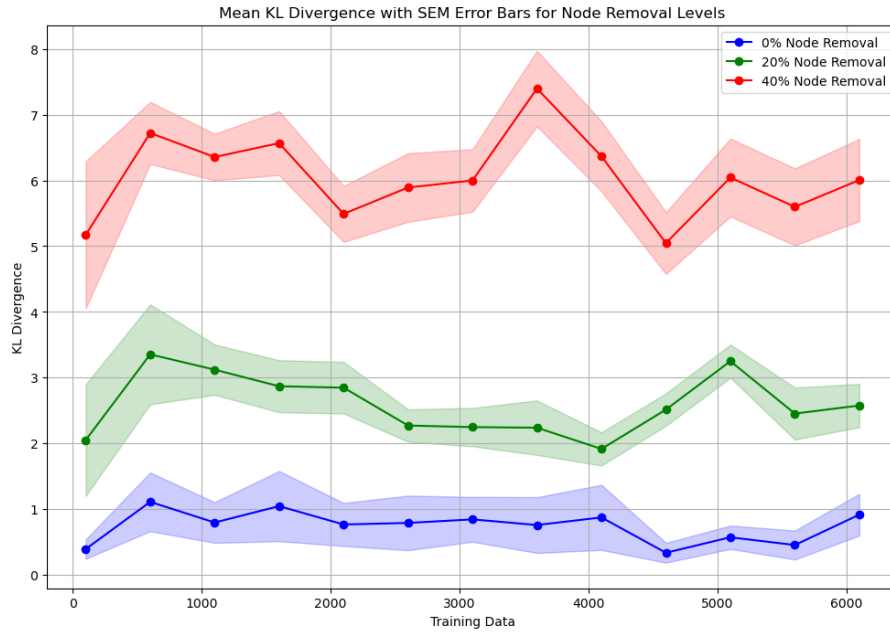


Figure 4.18: A plot of the training data versus the KL divergence for GTDs with 15 nodes using Hill Climb Search with BIC for structure learning with mutilation.

Figure 4.19 shows the relationship between training data size and the KL divergence for sparse Bayesian networks with 20 nodes. As the training data increases, the KL divergence decreases, indicating improved accuracy of the recovered structure. Sparse networks benefit from smaller datasets due to fewer dependencies, but the results also highlight a threshold beyond which additional data yields diminishing returns. This suggests that sparse structures are more efficient in scenarios with limited data availability.

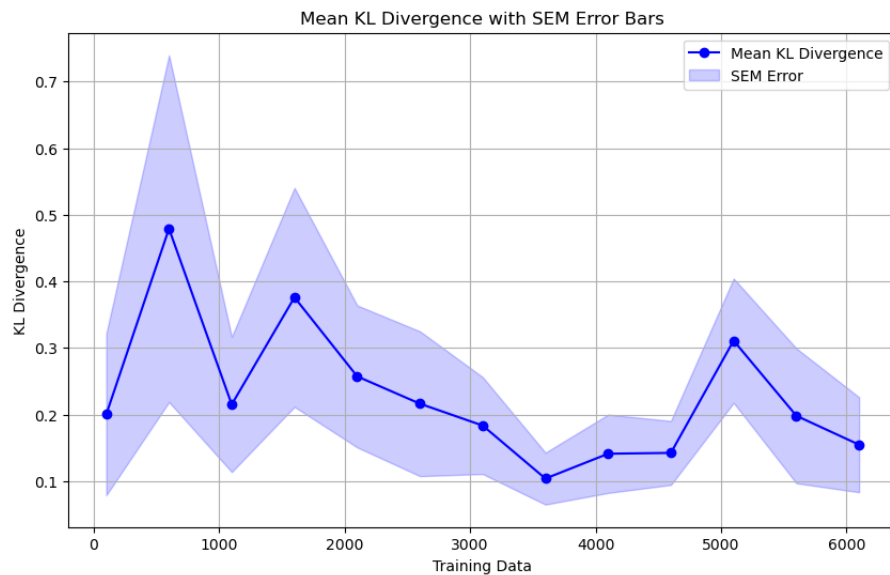


Figure 4.19: A plot of the training data versus the KL divergence for GTDs with 20 nodes using Hill Climb Search with BIC for scoring.

Figure 4.20 illustrates the impact of node removal on the KL divergence in sparse structures. The results show a sharper increase in KL divergence compared to Figure 4.19, reflecting the heightened sensitivity of sparse networks to node removal. However, recovery remains feasible with sufficient training data. This emphasizes the need to carefully identify critical nodes in sparse networks to mitigate resilience issues.

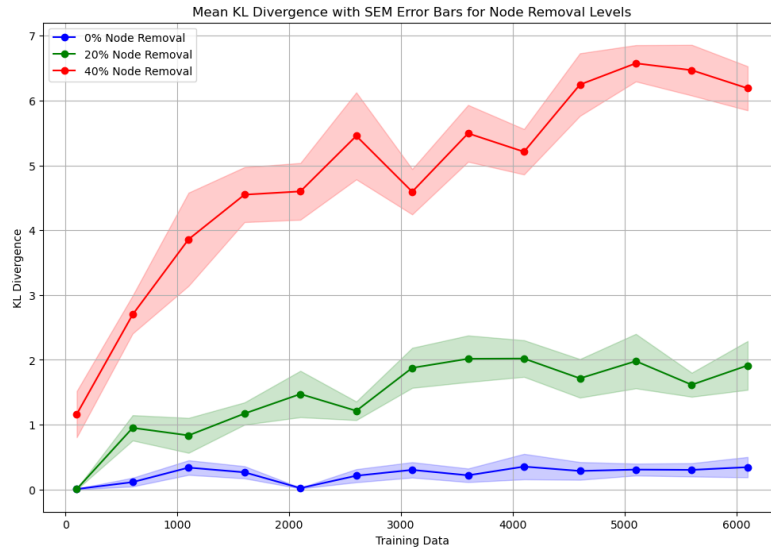


Figure 4.20: A plot of the training data versus the KL divergence for GTDs with 20 nodes using Hill Climb Search with BIC for structure learning with mutilation.

In Figure 4.21, the additional nodes increase structural complexity, slightly raising the initial KL divergence. However, with more training data, the KL divergence trends similarly to Figure 4.19, confirming that sparse networks with additional nodes can still stabilize effectively when provided adequate data.

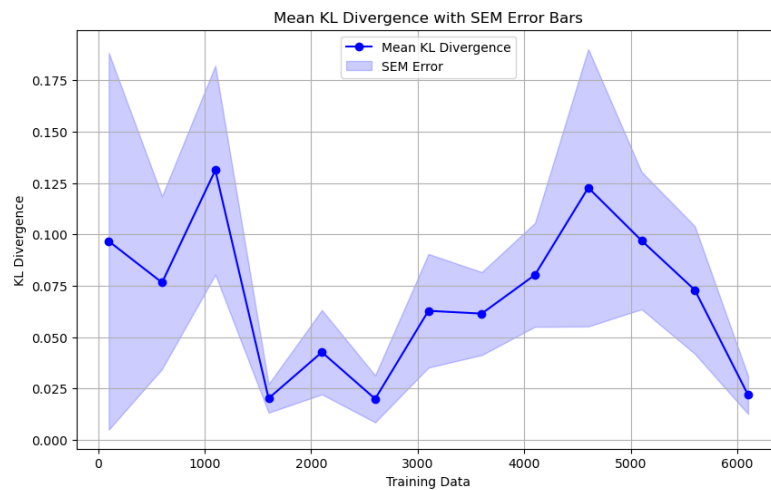


Figure 4.21: A plot of the training data versus the KL divergence for GTDs with 25 nodes using Hill Climb Search with BIC for scoring.

Figure 4.22 shows the effect of node removal on larger sparse structures. The results mirror the

observations in Figure 4.20 but exhibit slightly higher KL divergence due to increased node interactions. This underscores the importance of dataset size and structural simplicity when designing resilient sparse networks.

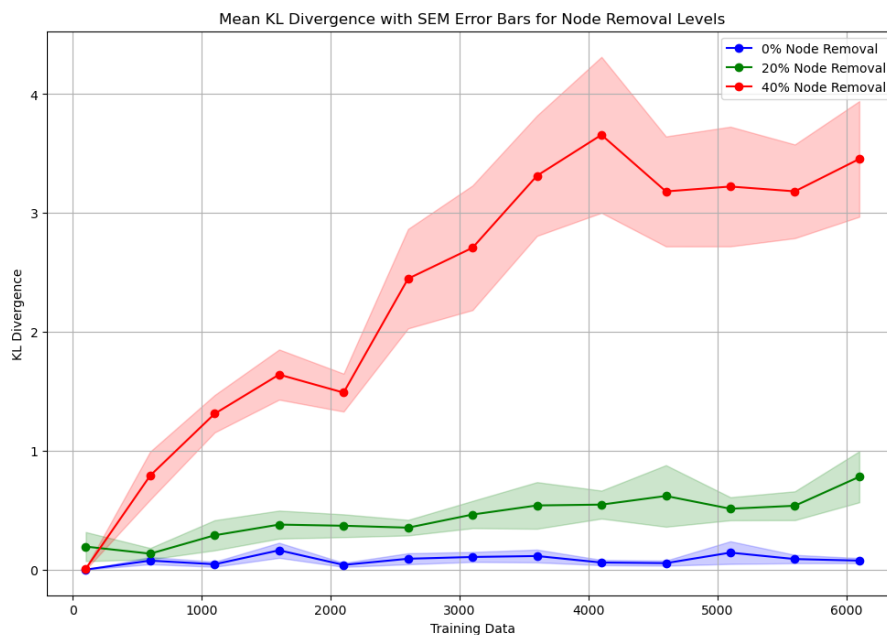


Figure 4.22: A plot of the training data versus the KL divergence for GTDs with 25 nodes using Hill Climb Search with BIC for structure learning with mutilation.

4.1.2.1 Discussion of observations and implications

Figure 4.2 depicts 10 sparse randomly generated DAGs with random Bernoulli CPDs and random edge orientations. The max n degree for each DAG is set at 2 with the edge probability being 0.1 for all sparse structures. Each DAG is treated as a different trial for each sample size. The DAGs are sampled and the produce samples that look like the samples that are depicted in Figure 4.13.

Structure learning is performed for each sample size, then the recovered models have their CPDs estimated with Bayesian estimation using the Dirichlet prior. The KL divergence is computed for all 10 trials, and then we take the mean KL divergence and we plot it with the error bars of that sample size given by the standard error of the mean (SEM). Figure 4.14 compares the performance of the different scoring methods when using the Hill Climb Search algorithm for structure learning. The Hill Climb Search is a heuristic method used in structure learning for Bayesian networks. Its aim is to search for the best network structure that fits a given dataset by iteratively modifying the structure and evaluating its score. Figure 4.14 has 3 scoring methods that are being compared and these are; the Bayesian Information Criterion (BIC), KIC/K2 score and the Bayesian structure scoring (BDScore).

The Bayesian Information Criterion is a model selection criterion based on the likelihood function, penalizing complex models to avoid over-fitting. The K2 score is a Bayesian scoring method designed specifically for discrete variables. It assumes a Dirichlet prior on the parameters and calculates the posterior probability of the network structure given the data. The Bayesian Dirichlet Score (BD) is a more general Bayesian scoring method, with variations such as BDeu which is the Bayesian Dirichlet equivalent uniform. It evaluates the posterior probability of a network structure by integrating out parameters, assuming a Dirichlet prior distribution for the conditional probabilities.

The search space grows exponentially with the number of nodes when performing model recovery with the Hill Climb Search so we have limited our experiments to 10, 15, 20, 25 nodes in the GTD DAGs. Figure 4.15 depicts the training data versus the KL divergence for GTDs with 10 nodes. Figure 4.16 compares the KL divergence of mutilated recovered models starting with no mutilation up to 40% node removal. Figures from 4.17 to 4.22 depict the same results but for more complex ground truth distributions. In contexts such as ATM networks, sparse structures efficiently represent critical dependencies with fewer connections; however, their limited connectivity may increase vulnerability to disruptions, requiring additional measures to enhance resilience and facilitate effective recovery.

4.1.3 Dense Structures

Dense structures in graphical models are characterized by a high number of edges relative to the number of nodes. In a dense graph, most pairs of nodes are connected directly or indirectly through many paths, resulting in complex inter-dependencies. These structures are highly connected as each node has many edges connecting it to other nodes, resulting in a high degree of connectivity. In terms of node removal, these structures have low sensitivity when removing a small number of nodes. However, removing highly connected nodes might still disrupt the network, especially if those nodes have significant influence. Structure learning in dense structures and inference are both computationally intensive due to large number of connections and inter-dependencies.

Figure 4.23 depicts the structure of 10 dense Directed Acyclic Graphs (DAGs), characterized by high interconnectivity between nodes. This interconnected nature provides robustness and redundancy, allowing these structures to handle disruptions better than sparse or tree-like configurations.

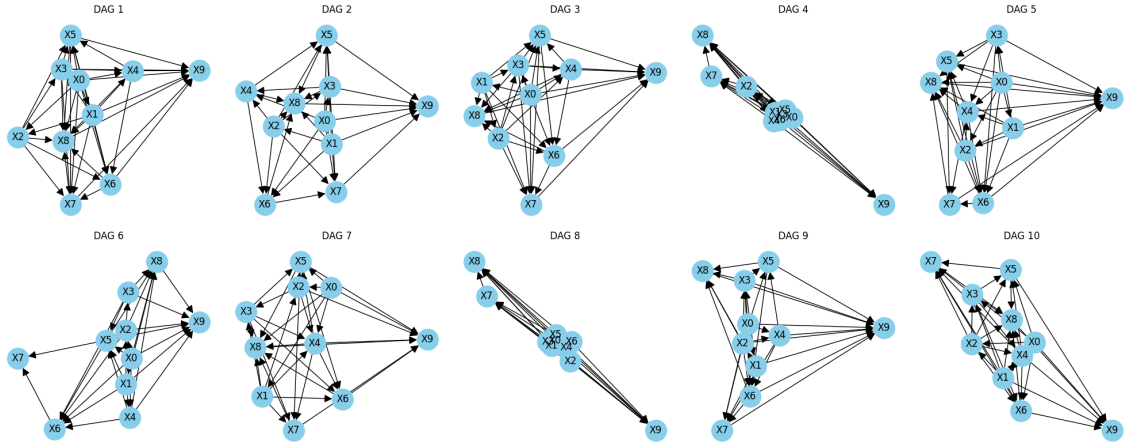


Figure 4.23: 10 randomly generated dense DAGs with random edge orientations and random Bernoulli CPDs assigned to each node.

Figure 4.24 presents the results of the KL divergence between the ground truth distribution and the recovered models for dense Directed Acyclic Graphs (DAGs) of 10 nodes. The experiments compare the performance of three scoring methods: Bayesian Information Criterion (BIC), Kullback Information Criterion (KIC), and Bayesian Dirichlet Score (BDScore), under the Hill Climb Search algorithm.

Figure 4.25 reveals that BIC consistently results in the lowest KL divergence compared to KIC and BDScore across varying amounts of training data. The BIC score's robustness in dense structures likely arises from its penalty for model complexity, which prevents overfitting. Meanwhile, KIC and BDScore exhibit slightly higher KL divergence values, indicating their sensitivity to the increased complexity of

dense networks. These findings underscore the importance of selecting an appropriate scoring method when working with dense Bayesian networks. The consistent performance of BIC suggests it is better suited for scenarios where dense dependencies exist among nodes, as it balances model fit and complexity. This insight is particularly relevant for domains like ATM networks, where high interconnectivity between transaction volumes is anticipated.

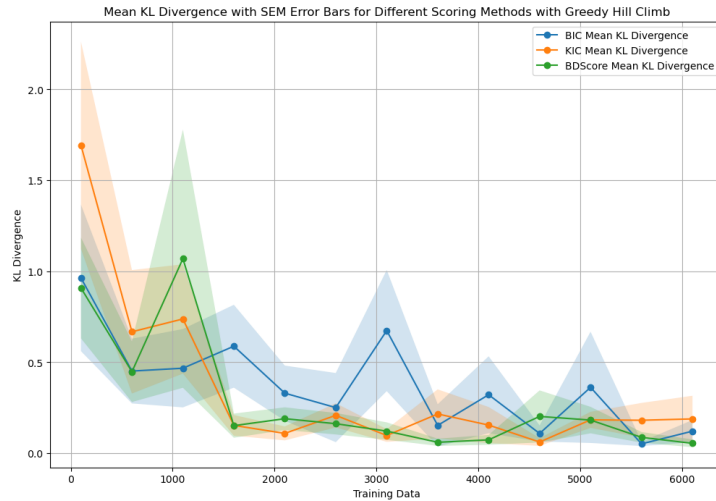


Figure 4.24: A plot of the training data versus the KL divergence for GTDs with 10 nodes comparing the BIC, KIC and BDScore for scoring while using the Hill Climb Search algorithm for structure learning.

Figure 4.25 shows the KL divergence as a function of the number of training samples used in structure learning. This analysis highlights how the amount of data impacts the accuracy of the recovered Bayesian network in approximating the ground truth. The figure illustrates that as the training data size increases, the KL divergence steadily decreases. These results emphasize the necessity of sufficient data for effective structure recovery in dense networks, where intricate dependencies require a more comprehensive dataset for accurate learning. The diminishing returns observed at higher sample sizes suggest a practical balance; acquiring enough data to achieve low KL divergence without expending unnecessary resources.

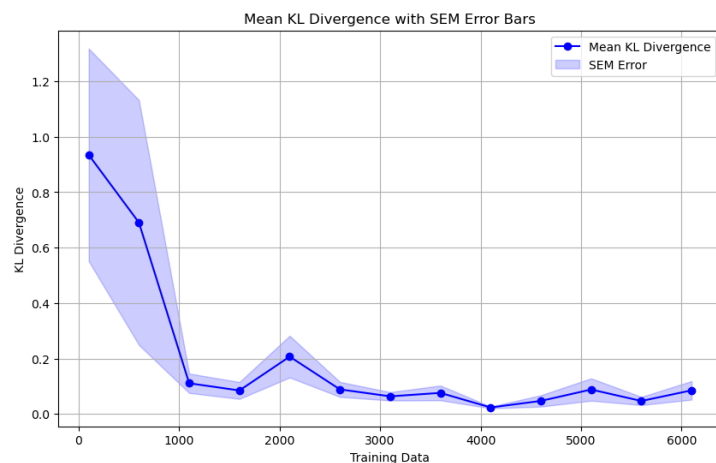


Figure 4.25: A plot of the training data versus the KL divergence for GTDs with 10 nodes using Hill Climb Search with BIC for scoring.

Figure 4.26 illustrates the KL divergence between the ground truth and the recovered models for a 10 node dense network, where nodes are removed incrementally. The figure reveals a sharp increase in KL divergence as more nodes are removed from the network. This trend highlights the difficulty in recovering the true structure when critical nodes and their associated edges are eliminated. Even with sufficient training data, the recovered models deviate significantly from the ground truth as the level of mutilation increases. This suggests that dense networks have a tipping point beyond which the structural integrity is severely compromised. The results underscore the fragility of dense networks to node removal, as their heavy inter-connectivity creates cascading effects when nodes are disrupted.

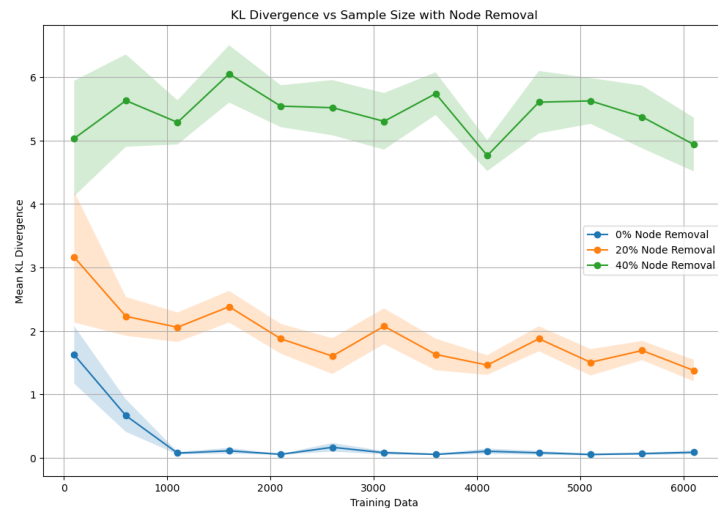


Figure 4.26: A plot of the training data versus the KL divergence for GTDs with 10 nodes using Hill Climb Search with BIC for structure learning with mutilation.

Figure 4.27 shows the KL divergence as a function of training dataset size for 15 node dense networks. By analyzing this figure, we gain insights into how training data requirements scale with network complexity. As with smaller networks, the KL divergence decreases as the training dataset size increases, indicating improved recovery of the true structure.

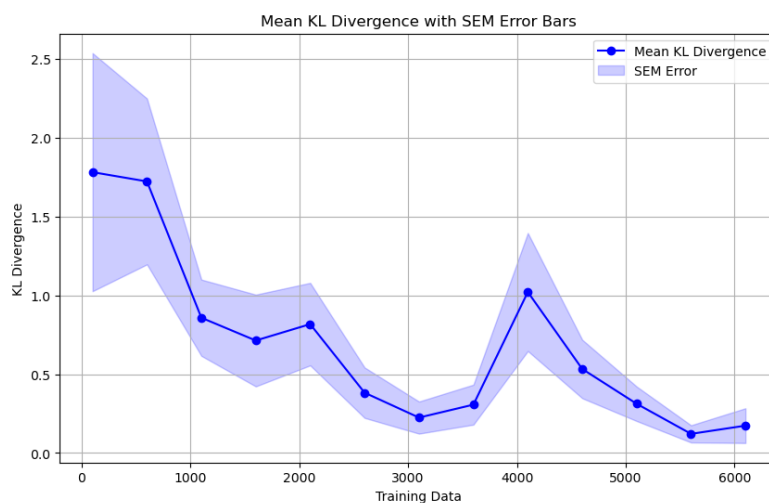


Figure 4.27: A plot of the training data versus the KL divergence for GTDs with 15 nodes using Hill Climb Search with BIC for scoring.

Figure 4.28 presents the KL divergence between the ground truth and recovered models for 15 node dense networks with incremental node removal. The figure indicates a steep rise in KL divergence as the proportion of removed nodes increases.

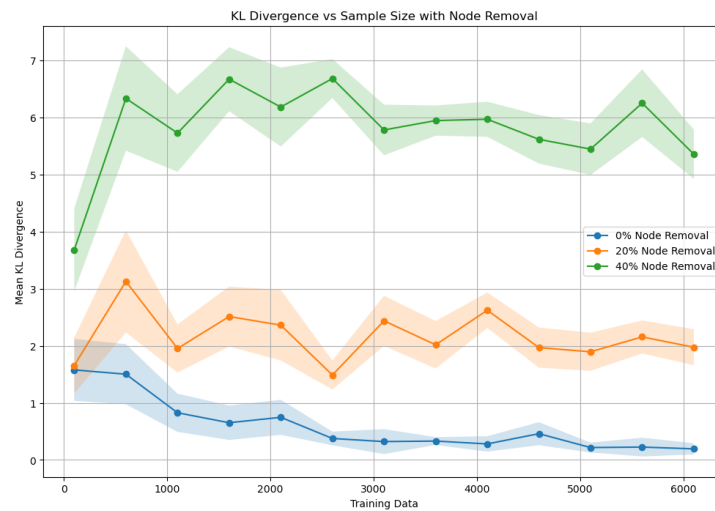


Figure 4.28: A plot of the training data versus the KL divergence for GTDs with 15 nodes using Hill Climb Search with BIC for structure learning with mutilation.

Figure 4.29 illustrates the relationship between training data size and KL divergence for a dense network with 20 nodes. The figure shows a consistent trend of decreasing KL divergence with increased training dataset sizes. However, compared to smaller networks, the KL divergence starts at higher values and decreases at a slower rate, indicating that larger networks require significantly more data to achieve similar recovery accuracy. The diminishing returns plateau appears earlier, highlighting the limits of additional training data in mitigating complexity for larger networks. The results suggest that as dense network size increases, data requirements grow exponentially to maintain recovery accuracy. For ATM networks, this emphasizes the importance of region-specific modeling with manageable network sizes to avoid data scalability issues.

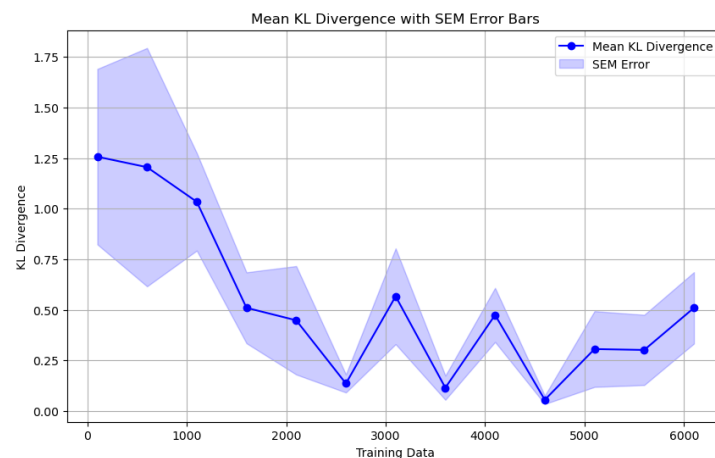


Figure 4.29: A plot of the training data versus the KL divergence for GTDs with 20 nodes using Hill Climb Search with BIC for scoring.

Figure 4.30 shows the KL divergence as a function of the percentage of nodes removed. The KL divergence increases steeply with higher levels of node removal. Compared to smaller networks, the rate of divergence is significantly accelerated, indicating that the loss of nodes and their dependencies in larger networks has a more profound effect on structure recovery.

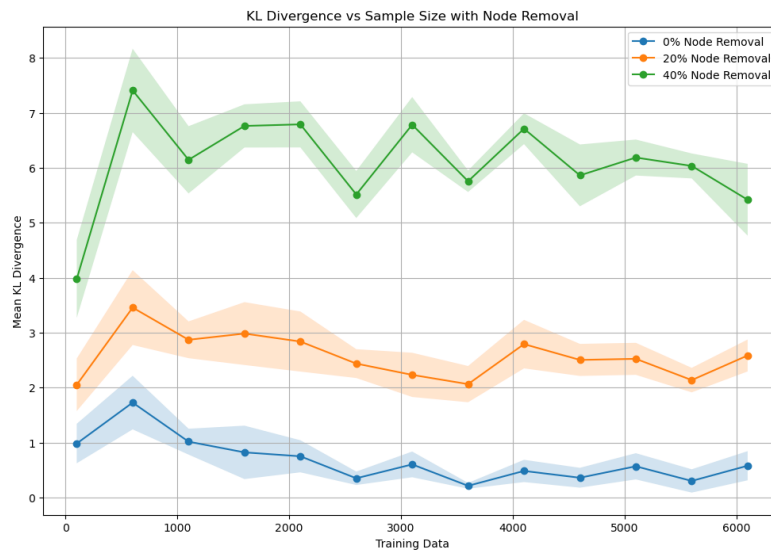


Figure 4.30: A plot of the training data versus the KL divergence for GTDs with 20 nodes using Hill Climb Search with BIC for structure learning with mutilation.

Figure 4.31 illustrates the relationship between training data size and KL divergence for 25 node dense networks. The KL divergence for 25 node networks starts at much higher values than for smaller networks, underscoring the challenges posed by increased complexity.

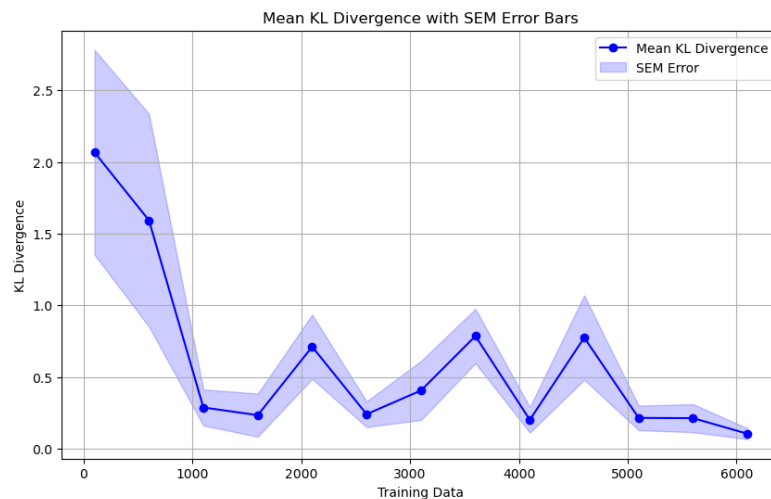


Figure 4.31: A plot of the training data versus the KL divergence for GTDs with 25 nodes using Hill Climb Search with BIC for scoring.

Figure 4.32 shows the KL divergence as a function of the percentage of nodes removed for a dense 25 node network. The figure reveals a pronounced increase in KL divergence with node removal, even at lower percentages of mutilation. Larger networks demonstrate higher susceptibility to node removal,

with the rate of divergence accelerating dramatically beyond a certain removal threshold. These results highlight the heightened vulnerability of larger dense networks to node removal, reinforcing the importance of redundancy and fault tolerance mechanisms. For ATM networks, these findings suggest that critical nodes must be identified and reinforced to mitigate cascading effects and ensure system resilience.

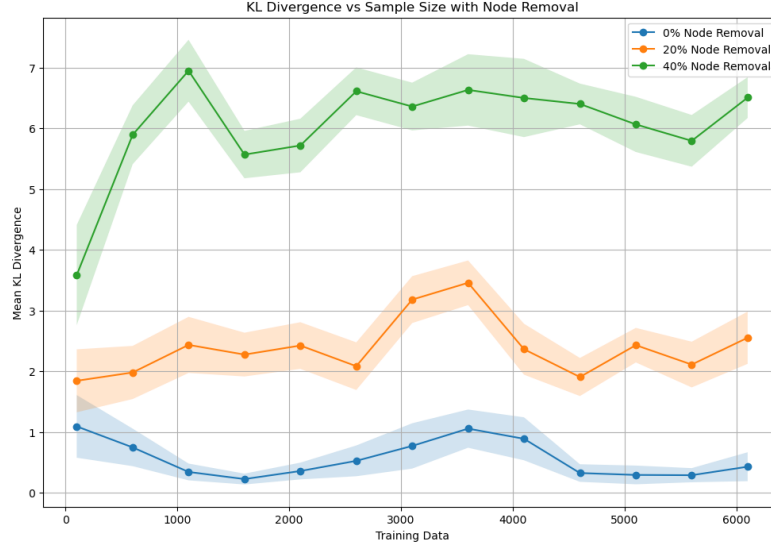


Figure 4.32: A plot of the training data versus the KL divergence for GTDs with 25 nodes using Hill Climb Search with BIC for structure learning with mutilation.

4.1.3.1 Discussion of observations and implications

Figure 4.22 depicts 10 dense randomly generated DAGs with random Bernoulli CPDs and random edge orientations. The max n degree for each DAG is set at 6 and the edge probability is set at 0.9 for all dense structures. Each DAG is treated as a different trial for each sample size. The experimental set up that is followed in subsection 4.1.2 is applied in this subsection. We have also limited our experiments to GTDs with DAGs that have 10, 15, 20 and 25 nodes. Figure 4.25 depicts the training data versus the KL divergence of mutilated recovered models starting with no mutilation up to 40% node removal. Figure 4.27 to Figure 4.30 depict the same results but for more complex ground truth distributions.

Dense Bayesian networks exhibit unique challenges in structure recovery and resilience to node removal due to their intricate dependency patterns. As network size increases, the KL divergence remains higher, even with significant training data, indicating that larger dense networks require exponentially more data to achieve accurate recovery. The effects of node removal are particularly severe in dense networks, with KL divergence increasing steeply as the percentage of removed nodes rises. Larger networks magnify these effects, demonstrating heightened fragility to disruptions. In terms of real world application for ATM networks, this suggests that larger datasets are crucial for modeling dense structures accurately, but diminishing returns must guide efficient data collection strategies. Dense networks require proactive design to maintain resilience against node removal, particularly for highly connected nodes. Subdividing large dense networks into smaller, region-specific clusters may help balance accuracy and scalability in modeling. In contexts such as ATM networks, dense structures offer extensive redundancy through numerous interconnections, which enhances robustness by providing multiple pathways for influence propagation; however, this complexity may necessitate larger datasets and more advanced techniques for accurate network recovery following disruptions.

4.1.4 Comparative Analysis and Synthesis Across Network Models

The effect of training data on recovery of Bayesian networks using structure learning differs significantly between tree-like, sparse and dense structures. When the training data increases, tree-like structures exhibit rapid stabilization with a clear hierarchical flow and few paths. The model quickly learns the relationships, and the KL divergence decreases predictably. This structure benefits from moderate amounts of data as there is minimal redundancy. In terms of sparse structures, these show initial stability due to limited connections.

The KL divergence stabilizes with sufficient data as fewer dependencies need to be learned. On the other hand, dense structures require substantial data to stabilize. Since these structures are highly connected, this introduces complexity and potential noise which in turn requires more datasets to capture the dependencies. Removing nodes from the recovered models impacts each structural configuration differently. Tree-like structures have a high initial sensitivity especially when we remove nodes that are central but this sensitivity decreases with more data as the dependencies are simple. The different experimental set ups for the GTDs starting from 10 nodes up to 150 nodes in the DAGs indicates that we are still able to recover these structures even when we go up to 40% node removal.

Sparse structures on the other hand, with less complex GTDs, we can see that structure learning attempts to recover the structures and dependencies for up to 20% of node removal but at 40% node removal it struggles to recover the models from the data. This suggests that a lot of dependencies get removed from the model with a higher degree of mutilation. The more complex the GTDs are, the harder it is for the structure learning algorithm to recover the model as more dependencies are lost. This problem is compounded for dense structures, node removal disrupts a lot of dependencies and to attempt to recover the models from the data becomes a computational challenge as more data is required and this results in an exponentially growing search space.

Practical applications would be modeling the influence of transaction volumes between ATMs. Tree-like networks can be chosen to represent hierarchical influences such as ATMs located in a region that is a rural area. These would not be highly connected but if a node (ATM) is critical it would present an issue if removed. Sparse structures would be ideal for identifying ATM hubs that are key but with minimal connections. Dense structures offer a more robust model for dense connected ATMs with urban areas being suitable locations.

4.2 Research Question 3: How can Insights from Bayesian Networks Improve the Operational Efficiency and Planning of ATM Networks?

Building on the insights gained from earlier experiments on Bayesian networks that include tree-like, sparse and dense structures, this section applies the methodology to a real-world use case for ATM networks. The goal is to explore how Bayesian networks can model the influence relationships among ATMs and how disruptions in the network affect transaction volumes and dependencies. By combining theoretical findings with practical data, this section demonstrates the applicability of Bayesian modeling in operational decision-making.

Figure 4.33 shows temporal variations in transaction volumes, with notable peaks at month-ends and dips during holidays. These patterns suggest a strong temporal dependency, indicating that ATMs in high-demand locations require additional planning for cash replenishment. The figure depicts ATM transaction volumes for a period of 4 years.

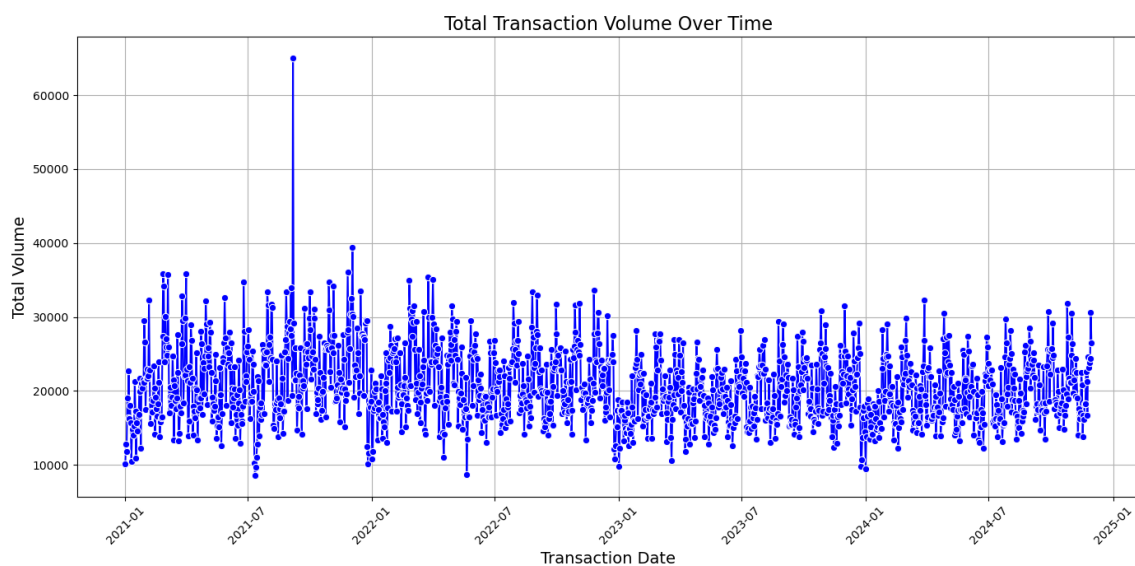


Figure 4.33: A plot of the total transaction volumes for 156 ATMs starting from January 2021 until November 2024.

The heat map in Figure 4.34 reveals clustering in transaction volumes among ATMs in close proximity. This implies spatial dependencies, where transaction volumes in one ATM correlate with nearby machines, highlighting the importance of spatial optimization in ATM network planning.

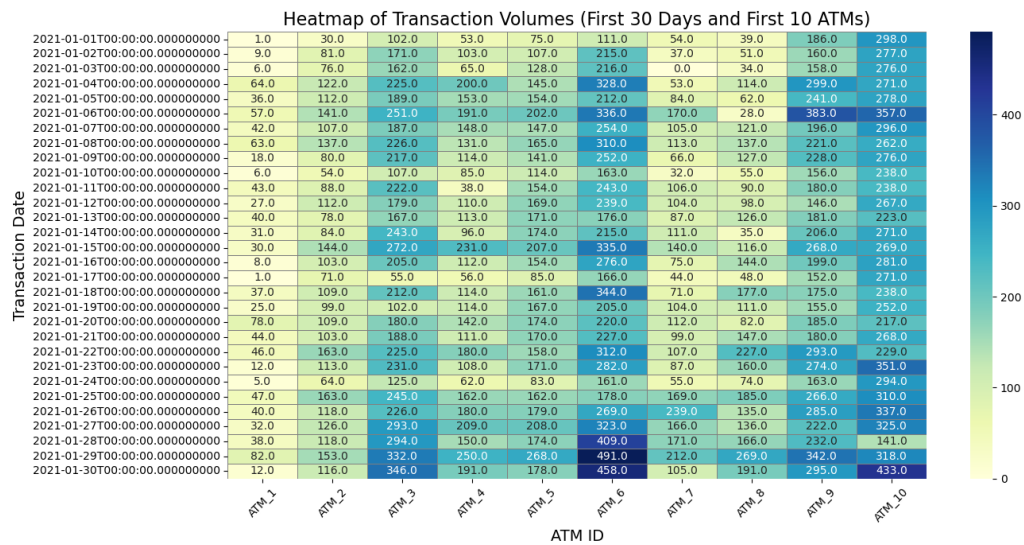


Figure 4.34: A heatmap of transaction volumes for first 30 days for the first 10 ATMs.

The box plot in Figure 4.35 visualizes the transaction volume distributions for the first five ATMs. The plot highlights variability in transaction volumes across ATMs, with some exhibiting higher median transactions and others showing greater variability or outliers. This observation suggests that certain ATMs handle a significantly higher load, potentially due to their location or customer demographics. Implications for ATM network management include identifying and reinforcing high-demand ATMs to ensure operational continuity and optimizing cash replenishment schedules for machines with high variability to minimize downtime.

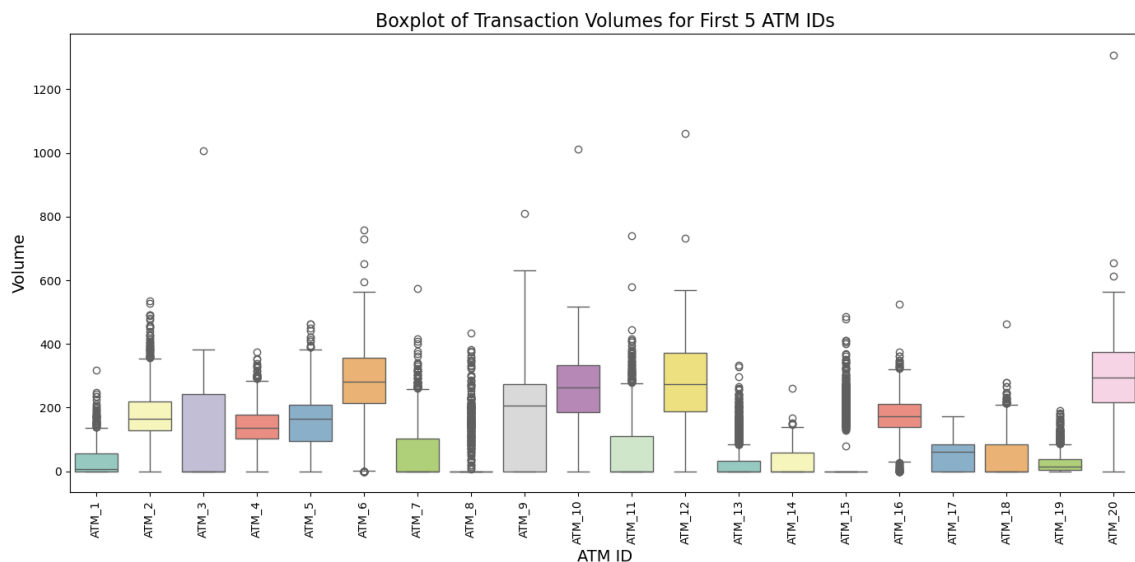


Figure 4.35: A Box Plot of transaction volumes for the first 5 ATMs.

Figure 4.36 presents the distribution of transaction volumes for a single ATM (ATM 1). The distribution shows a peak around specific transaction values, reflecting customer withdrawal behavior. This regularity can guide cash replenishment policies to match predictable demand patterns, reducing both shortages and excess cash storage.

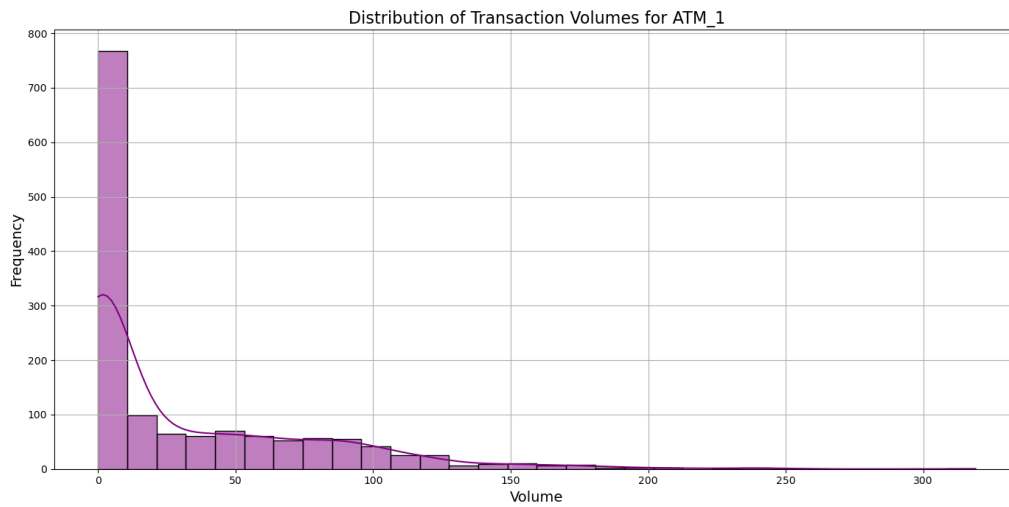


Figure 4.36: A plot of the distribution of transaction volumes for ATM 1.

Figure 4.37 illustrates the quantized transaction data for a subset of ATMs. By grouping transaction amounts into discrete bins, the data highlights general trends and outliers. This quantization is crucial for applying Bayesian network models, as it simplifies the structure and makes dependency modeling more tractable and similar to the approach taken for the synthetic data part of our study which uses Bernoulli CPDs.

Top 10 entries of the dataset:									
	Transaction_Dt	ATM_1	ATM_2	ATM_3	ATM_4	ATM_5	ATM_6	ATM_7	ATM_8 \
0	2021/01/01	0	0	0	0	0	0	0	0
1	2021/01/02	0	0	1	0	0	1	0	0
2	2021/01/03	0	0	1	0	1	1	0	0
3	2021/01/04	0	1	1	1	1	1	0	0
4	2021/01/05	0	0	1	1	1	1	0	0
5	2021/01/06	0	1	1	1	1	1	1	0
6	2021/01/07	0	0	1	1	1	1	0	1
7	2021/01/08	0	1	1	1	1	1	0	1
8	2021/01/09	0	0	1	0	1	1	0	1
9	2021/01/10	0	0	0	0	0	1	0	0

Figure 4.37: A snapshot of the quantized ATM transaction data.

The Bayesian network in Figure 4.38 shows the inferred dependencies among the first 20 ATMs. Nodes represent ATMs, while directed edges indicate influence relationships. The results reveal clusters of highly interconnected ATMs, suggesting areas of shared demand and potential redundancy. Banks can use this information to optimize operations and resource allocation.

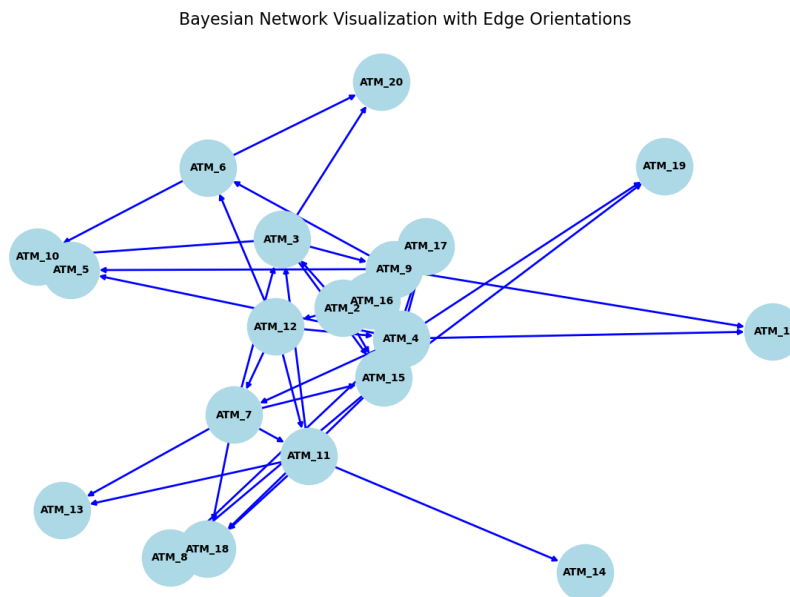


Figure 4.38: A recovered Bayesian Network for the first 20 ATMs.

Overlaying the Bayesian network on a GIS map (Figure 4.39) provides spatial context to the inferred dependencies. This visualization identifies geographical clusters of ATMs and helps detect areas vulnerable to disruptions. Such insights can improve strategic planning, including redundancy and service coverage.

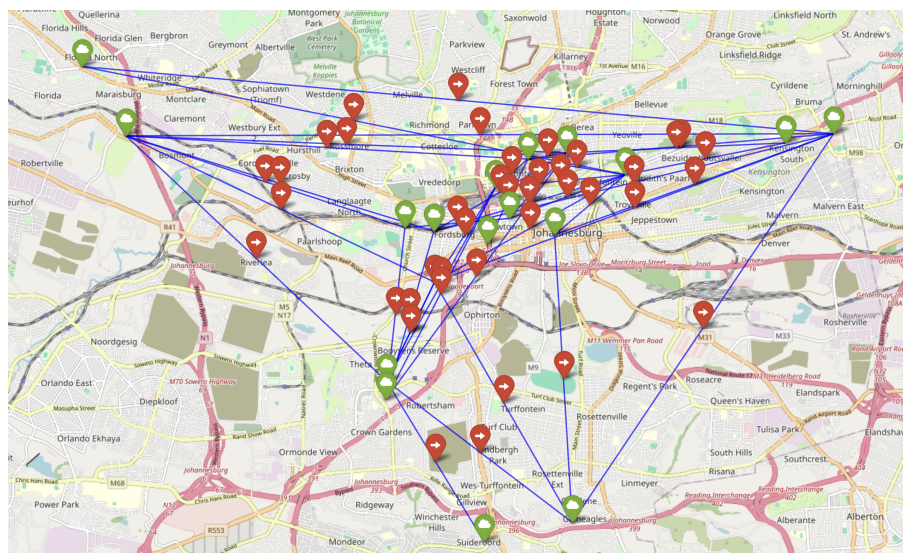


Figure 4.39: A recovered Bayesian Network for the first 20 ATMs overlaid on a GIS map with their longitude and latitude.

Figure 4.40 evaluates the model's performance using metrics such as precision, recall, and accuracy for three specific ATMs. High performance across these metrics validates the network's ability to capture transactional relationships accurately. This confirms the model's utility in predicting the impact of network changes.

```
Precision for ATM_15: 0.49056603773584906
Recall for ATM_15: 0.6046511627906976
Accuracy for ATM_15: 0.56
Precision for ATM_3: 0.4791666666666667
Recall for ATM_3: 0.46938775510204084
Accuracy for ATM_3: 0.49
Precision for ATM_20: 0.5178571428571429
Recall for ATM_20: 0.5686274509803921
Accuracy for ATM_20: 0.51
```

Figure 4.40: Evaluation metrics applied on 3 ATMs.

Figure 4.41 shows a reduced Bayesian network after removing ATM 15. The updated structure highlights how dependencies are reallocated among remaining ATMs, revealing the resilience of the network to disruptions. Identifying such shifts can guide contingency plans for maintaining service continuity.

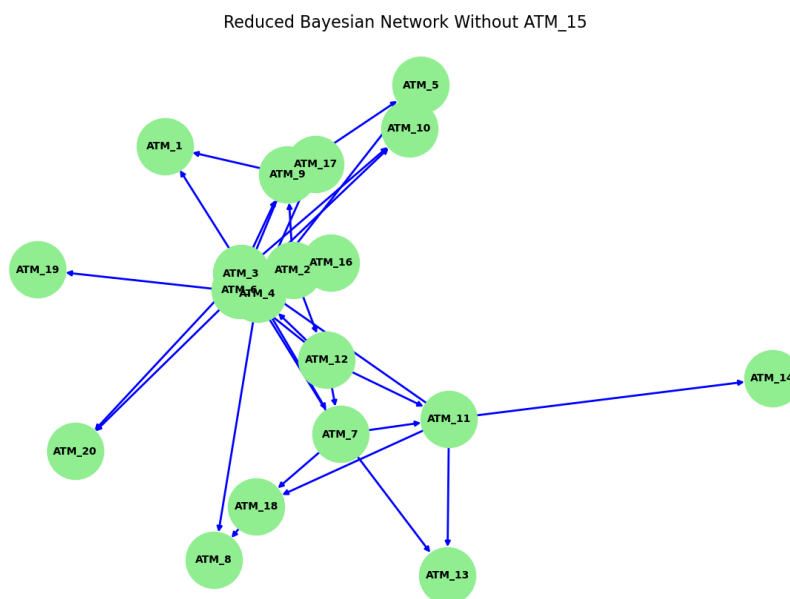


Figure 4.41: Reduced graph structure with the removal of ATM 15

When mapped geographically (Figure 4.42), the reduced network shows localized impacts of removing ATM 15. The visualization emphasizes the need for redundancy in high-demand areas and helps prioritize recovery efforts post-disruption.

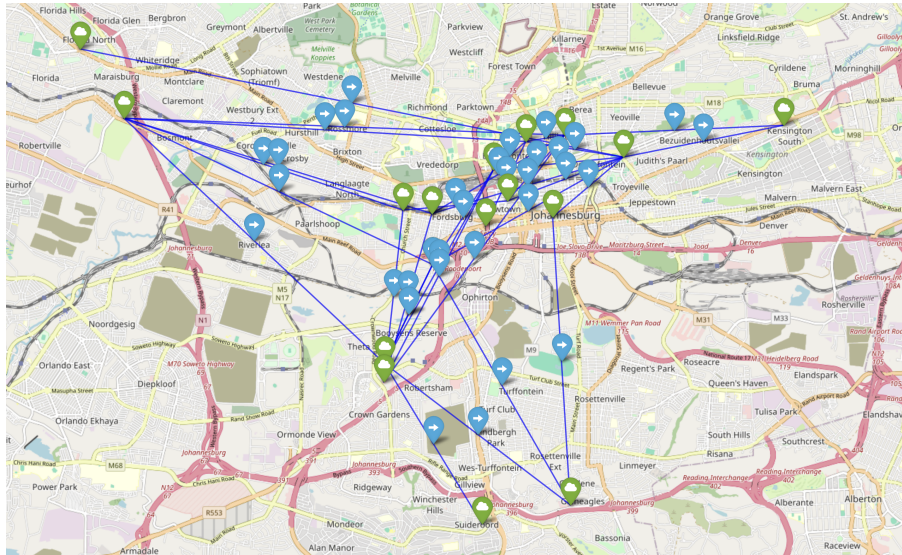


Figure 4.42: Reduced graph structure with the removal of ATM 15 overlaid on a GIS map.

Figure 4.43 depicts the evaluation metrics after reducing the network by removing specific nodes, compared to the original recovered network shown in Figure 4.40. In the original recovered network (Figure 4.40), accuracy remains high due to the complete structure's ability to capture dependencies effectively. However, in the reduced network (Figure 4.43), accuracy drops noticeably, reflecting the missing connections and degraded ability to model the data correctly. Precision also declines in the reduced network as the mutilation removes specific node-to-node relationships, increasing the likelihood of incorrect associations. The original network shows superior precision, particularly in identifying accurate parent-child dependencies between nodes. Recall is significantly affected by the reduction. With fewer nodes and missing edges, the reduced network fails to capture many true relationships present in the original ground truth. The recall in Figure 4.43 is substantially lower than in Figure 4.40, emphasizing the reduced network's inability to account for all valid dependencies.

The reduction in accuracy, precision, and recall indicates that node removal significantly impacts the recovered network's ability to represent the ground truth structure. Dense networks, in particular, suffer more pronounced degradation due to their reliance on intricate interconnections, which are disrupted when nodes are removed.

```
Precision for ATM_3: 0.5
Recall for ATM_3: 0.42857142857142855
Accuracy for ATM_3: 0.51
Precision for ATM_20: 0.4418604651162791
Recall for ATM_20: 0.4418604651162791
Accuracy for ATM_20: 0.52
```

Figure 4.43: Evaluation metrics on two ATMs after the graph is reduced.

The Bayesian network in Figure 4.44 shows the inferred dependencies among the 156 ATMs. Nodes represent ATMs, while directed edges indicate influence relationships. The results reveal clusters of highly interconnected ATMs which are similar to the dense network structures from the synthetic data, suggesting areas of shared demand and potential redundancy. Banks can use this information to optimize operations and resource allocation.

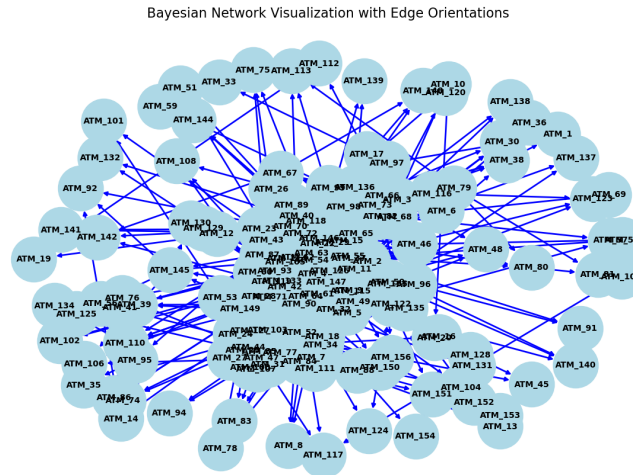


Figure 4.44: A recovered Bayesian Network for the first 156 ATMs.

Overlaying the Bayesian network on a GIS map (Figure 4.45) provides spatial context to the inferred dependencies. This visualization identifies geographical clusters of ATMs and helps detect areas vulnerable to disruptions. Such insights can improve strategic planning, including redundancy and service coverage.

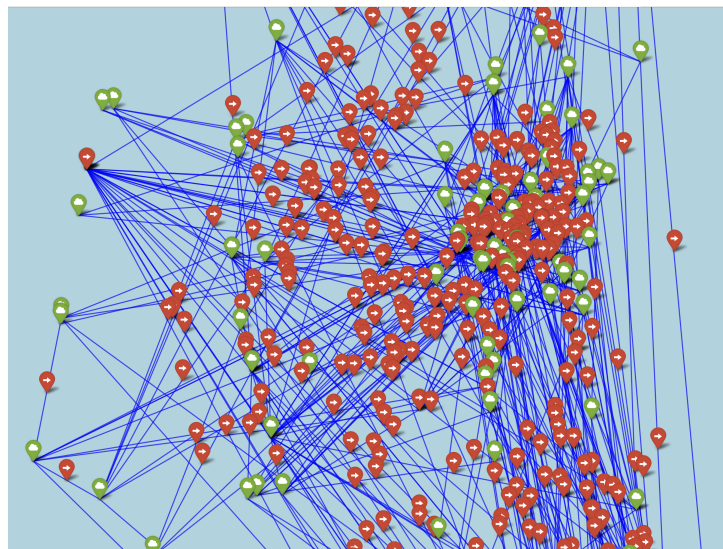


Figure 4.45: Snapshot of a recovered Bayesian Network for 156 ATMs.

Figure 4.46 evaluates the model's performance using metrics such as precision, recall, and accuracy for four specific ATMs before the removal of ATM 15. High performance across these metrics validates

the network’s ability to capture transactional relationships accurately. This confirms the model’s utility in predicting the impact of network changes.

```

--- Metrics BEFORE Removal of ATM_15 ---
Metrics for ATM_62:
Precision: 0.45898839215686275
Recall: 0.5348837285382325
Accuracy: 0.52
Metrics for ATM_105:
Precision: 0.4
Recall: 0.5454545454545454
Accuracy: 0.44
Metrics for ATM_3:
Precision: 0.5
Recall: 0.5869565217391385
Accuracy: 0.54
Metrics for ATM_20:
Precision: 0.5518284881632653
Recall: 0.5192387692387693
Accuracy: 0.53
Removing ATM_15 impacts the following ATMs: ['ATM_65', 'ATM_62', 'ATM_105', 'ATM_137', 'ATM_92', 'ATM_80', 'ATM_98', 'ATM_111', 'ATM_146']

```

Figure 4.46: Evaluation metrics applied on ATM 62, ATM 105 and ATM 3 and ATM 20.

Figure 4.46 shows a reduced Bayesian network after removing ATM 15. The updated structure highlights how dependencies are reallocated among remaining ATMs, revealing the resilience of the network to disruptions. Identifying such shifts can guide contingency plans for maintaining service continuity.

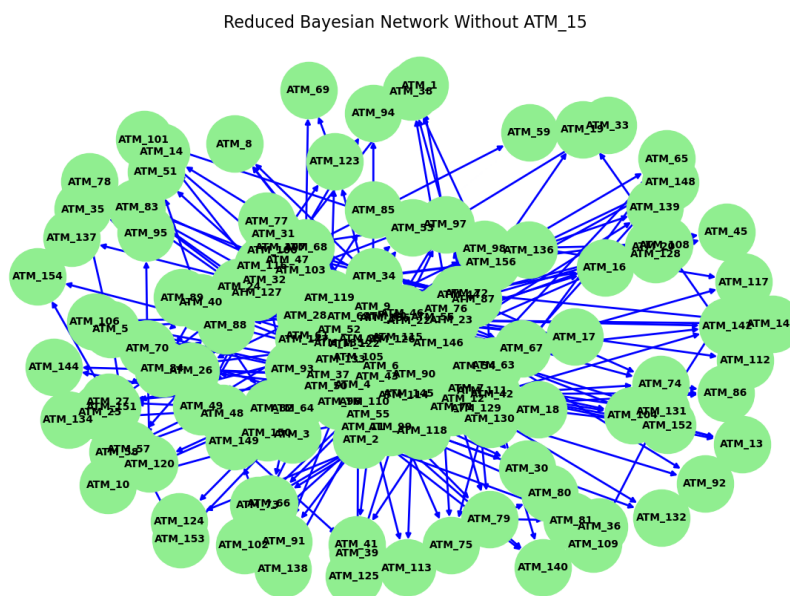


Figure 4.47: Reduced graph structure with the removal of ATM 15.

When mapped geographically (Figure 4.47), the reduced network shows localized impacts of removing ATM 15. The visualization emphasizes the need for redundancy in high-demand areas and helps prioritize recovery efforts post-disruption. The recovered structure exhibits properties of a dense structure that models the underlying ATM data.

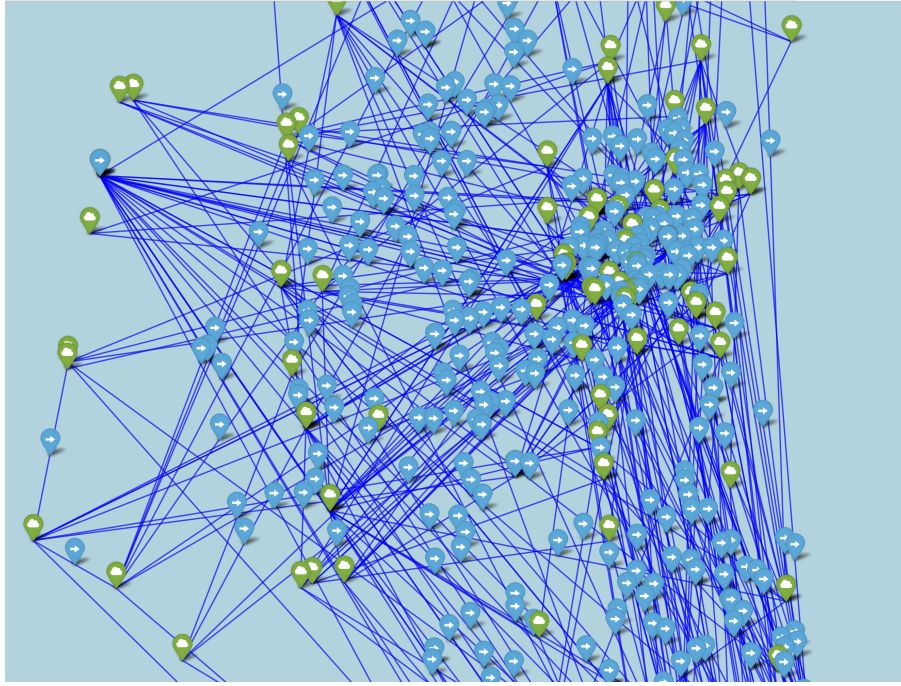


Figure 4.48: Reduced graph structure with the removal of ATM 15.

Figure 4.49 evaluates the model's performance using metrics such as precision, recall, and accuracy for four specific ATMs after the removal of ATM 15. High performance across these metrics validates the network's ability to capture transactional relationships accurately. This confirms the model's utility in predicting the impact of network changes.

```

--- Metrics AFTER Removal of ATM_15 ---
Metrics for ATM_62:
  Precision: 0.52
  Recall:    0.49056603773584906
  Accuracy: 0.49
Metrics for ATM_105:
  Precision: 0.5370370370370371
  Recall:    0.5087719298245614
  Accuracy: 0.47
Predicted transaction volume for ATM_15 next month: 92.66666666666666
Predicted transaction volume for ATM_3 next month: 78.12121212121211
Predicted transaction volume for ATM_20 next month: 69.46969696969697

```

Figure 4.49: Evaluation metrics applied on ATM 62, ATM 105 and ATM 3 and ATM 20.

The evaluation metrics provide a comprehensive insight into how the network's predictive performance shifts before and after the removal of a key node ATM_15. Initially, the model reports baseline metrics (precision, recall, and accuracy) for several ATMs such as ATM_62, ATM_105, ATM_3, and ATM_20. These metrics serve as a reference point, where precision indicates the fraction of predicted high-transaction events that are truly observed, recall represents the proportion of actual high-transaction events captured by the model, and accuracy reflects the overall rate of correct predictions.

Upon removing ATM_15, the network structure is perturbed, and a new evaluation shows significant changes in these metrics. For instance, ATM_62's precision increases from approximately 0.45 to 0.52, although its recall declines from about 0.53 to 0.49. Similarly, ATM_105 experiences improvements in both precision and recall. These changes indicate that the removal of ATM_15 has a cascading effect on the model's performance, highlighting that ATM_15 was a critical node in the network. The model identifies the impacted ATMs, namely, ATM_65, ATM_62, ATM_105, ATM_137, ATM_92, ATM_80,

ATM_90, ATM_111, and ATM_146 as those whose performance metrics shift notably after ATM_15 is removed. This identification is achieved by re-evaluating the conditional dependencies and influence relationships in the Bayesian network; nodes with altered inference performance or significant metric changes are flagged as impacted.

Graphical models, particularly Bayesian networks developed through structure learning, are powerful because they encapsulate complex interdependencies among nodes based on historical transaction data. This capability allows them not only to forecast current performance metrics but also to simulate “what if” scenarios. For example, after the removal of ATM_15, the model is still able to forecast next-month transaction volumes for several ATMs, predicting, for instance, transaction volumes of approximately 92.67 for a hypothetical continuation of ATM_15’s operations, 78.12 for ATM_3, and 69.47 for ATM_20. This demonstrates that even when the structure of the network is altered, the model retains its ability to generate useful forecasts.

The practical implications of these findings are significant. Decision-makers can use such evaluations to identify vulnerable points within the ATM network; by simulating the removal of other critical nodes, they can develop contingency plans to redistribute load or reinforce specific network segments. Moreover, the forecasts provide actionable insights for targeted maintenance and strategic expansion, if an ATM is scheduled for maintenance, the model can predict which neighboring ATMs might experience increased transaction volumes, enabling preemptive resource reallocation. Additionally, the model’s ability to simulate different scenarios, such as the impact of adding new ATMs or rerouting transaction flows, offers a powerful tool for long-term planning and load balancing, ensuring network resilience even under disruptive conditions.

In summary, the evaluation metrics before and after the removal of ATM_15 reveal how critical nodes influence the overall network performance. The model’s identification of impacted ATMs and its robust forecasting capabilities underscore the power of graphical models and structure learning. These insights not only validate the model’s performance but also enable a range of “what if” scenarios that can be leveraged to enhance operational resilience, optimize resource allocation, and guide strategic decision-making in complex financial networks.

4.2.1 Comparing Original and Reduced Networks

In this subsection, we analyze the effects of node removal on the Bayesian networks recovered from ATM transaction data by comparing the original, unaltered networks with their reduced counterparts and extending these observations to include insights from synthetic data experiments to establish generalizable conclusions about the resilience and adaptability of different network structures. The removal of nodes alters the structure of the Bayesian network, leading to significant changes in its probabilistic dependencies; in the original network, each ATM node maintains a set of conditional dependencies that reflect transaction volume correlations with neighboring ATMs, and these dependencies are learned through structure learning algorithms and validated using probabilistic queries.

When nodes are removed, nodes directly connected to a removed ATM must reorganize their dependency relationships, often forming new indirect dependencies or losing certain influences altogether, while the removal disrupts the network’s Markov properties and may introduce new conditional independencies, thereby altering the original influence structure. A key measure for evaluating this impact is the shift in conditional probability distributions, as evidenced by an increased Kullback-Leibler divergence between the probability distributions of the original and reduced networks, indicating that structure loss directly affects the reliability of probabilistic inference; in addition, Bayesian networks with a tree-like

structure tend to stabilize faster after node removal due to their inherent sparsity, whereas sparse networks show higher variability and dense networks, despite their robustness, require larger datasets for meaningful recovery.

Insights from synthetic data experiments reinforce these observations: tree-like networks demonstrate resilience to node removal through limited redundant connections and quick stabilization of probability distributions, mirroring the behavior observed in ATM networks where transaction flows redistribute predictably, while sparse networks exhibit high sensitivity with greater fluctuations in divergence and dense networks retain structural integrity through redundancy in dependencies, much like highly connected ATM networks where multiple pathways can absorb disruptions.

The comparison between the original recovered network (Figure 4.40) and the reduced network (Figure 4.43) highlights the significant effects of node removal on evaluation metrics. The original network demonstrates higher accuracy, precision, and recall, as it retains the complete set of nodes and their dependencies. In contrast, the reduced network exhibits a sharp decline in these metrics, underscoring the loss of structural information and dependencies due to node removal.

These findings have significant practical implications, as high-traffic ATMs serve as critical nodes whose removal leads to substantial transaction redistribution, prompting banks to use Bayesian network analysis to prioritize maintenance and security measures, adjust cash replenishment strategies, and develop adaptive routing strategies that minimize customer inconvenience during ATM outages. Overall, the comparison of original and reduced networks demonstrates that Bayesian networks effectively capture the influence relationships in ATM transaction systems, and findings from synthetic data provide further validation by revealing that different network structures exhibit varying degrees of resilience to node removal, an insight crucial for optimizing financial infrastructure and reinforcing the value of probabilistic graphical models in real-world applications.

There were limitations to our study based on the assumptions we made about our variables and the set of distributions that we used for the ground truth distribution. In this case, we made use of the Bernoulli distribution to generate the conditional probability distributions (CPD) which imposed a limitation to our study as ATMs are continuous variables and they need to be modeled with continuous variables because they model transactions. Another limitation to our study was the choice of structural configurations for our Ground Truth Distributions.

4.3 Conclusion

This chapter explored model recovery from data and the impact of node removal on recovered Bayesian networks. This was performed on 3 structural configurations and these being; tree-like, sparse and dense structures. Through the generation of synthetic data and structure learning, we demonstrated that each structure responds differently to mutilation and benefits distinctively with increased data. Tree-like structures demonstrated rapid stabilization of the KL divergence with increased data. They are easily interpretable and efficient but sensitive to central node removal. Sparse structures due to their limited connectivity, they exhibited initial higher vulnerability but with sufficient data they showed that they are suitable for systems with few dependencies.

Dense structures provided robustness but their complexity demands larger datasets for accurate learning and stabilization. These insights have significant practical applications for modeling the influence of transaction volumes between ATMs. Depending on data availability and resilience needs, financial institutions can choose appropriate network structures for their modeling needs. The findings highlight the importance of considering both structural complexity and the data volume. This is when we are applying Bayesian networks to real-world problems, ensuring resilient and accurate models for

dynamic environments.

Cross-structure comparisons highlight trade-offs, aiding decision-making in network design. The ATM model demonstrates the utility of Bayesian networks in real-world scenarios, offering actionable insights for financial institutions. Sparse and dense structures respond differently to node removal, with dense structures exhibiting greater resilience while tree-like structure prove to be more resilient if critical nodes are not removed, this addresses the first and second research questions. Structural complexity significantly influences recovery, with dense structures requiring larger datasets but offering higher robustness and this addresses research question two. Lastly, Bayesian modeling provides actionable insights for optimizing ATM operations for a practical use case which uses real transaction data, demonstrating practical value in financial systems.

Chapter 5

Conclusion and Future Work

This study set out to investigate the effects of node removal on Bayesian network resilience, starting with a rigorous mathematical formulation of the problem. By defining the joint probability distributions, conditional dependencies, and influence structures within the network, we established a clear framework for quantifying the impact of node removal. The mathematical models provided a solid foundation for understanding how disruptions propagate through the network and how structural changes affect the overall system behavior.

This research then aimed to model and analyze the influence of transaction volumes between Automated Teller Machines (ATMs) using Bayesian learning. Our primary objective was to explore how the joint probability distribution and influence structure among ATMs evolve when a subset of ATMs is removed. Through this study, we developed probabilistic graphical models to represent the relationships between ATMs and their evaluated robustness under various structural modifications. We successfully applied Bayesian networks to model the dependencies between nodes and showed how this can be applied to real-world data. Through generating synthetic data from different structural configurations for the ground truth distributions, we demonstrated how removing a node or a subset of nodes impacts the overall network.

Our approach provides new insights that can be applied into optimizing ATM placement and ensuring resilience against failures due to bombings or maintenance. Three types of structural configurations were explored for the ground truth distributions and these are; tree-like, sparse and dense structures. Our findings indicate tree-like structures have rapid stabilization with increased data, which is due to their simpler hierarchical connections. Sparse structures demonstrated higher vulnerability initially due to their limited connectivity but they stabilized with sufficient data. Dense structures on the other hand exhibited robustness to their redundant paths but require more data to maintain accuracy after node removal.

We utilized the Kullback-Leibler (KL) divergence to measure deviations between the original and modified structures. This approach highlighted the extent to which the network's predictive capability is affected by the removal of nodes. Additionally, an exact inference approach which is variable elimination was applied to validate recovered models and make comparisons to the original distribution. The results of this research have significant implications for the banking industry, particularly in optimizing ATM networks. Security and efficiency can be enhanced by understanding how transaction volumes shift when certain ATMs are deactivated, banks can better plan cash replenishment and predict customer behavior, thereby reducing operational risk.

In terms of resource allocation, the insights gained can assist financial institutions in strategic

decision-making, such as determining the optimal locations for new ATMs or deciding which under performing ATMs to remove. While this study made significant contributions, it also faced limitations. The scope of the structural complexity was limited due to computational constraints. Expanding the analysis to include more complex or dynamic structures could provide additional insights. The use of Bernoulli distributions for Conditional Probability Distributions may not fully capture the continuous nature of ATM transaction volumes. Incorporating Gaussian or other continuous distributions could improve accuracy.

Future research could extend the current static models to dynamic Bayesian networks (DBNs) to capture temporal changes in transaction volumes more effectively. The below figure demonstrates a dynamic Bayesian network for a temporal set up of the problem.

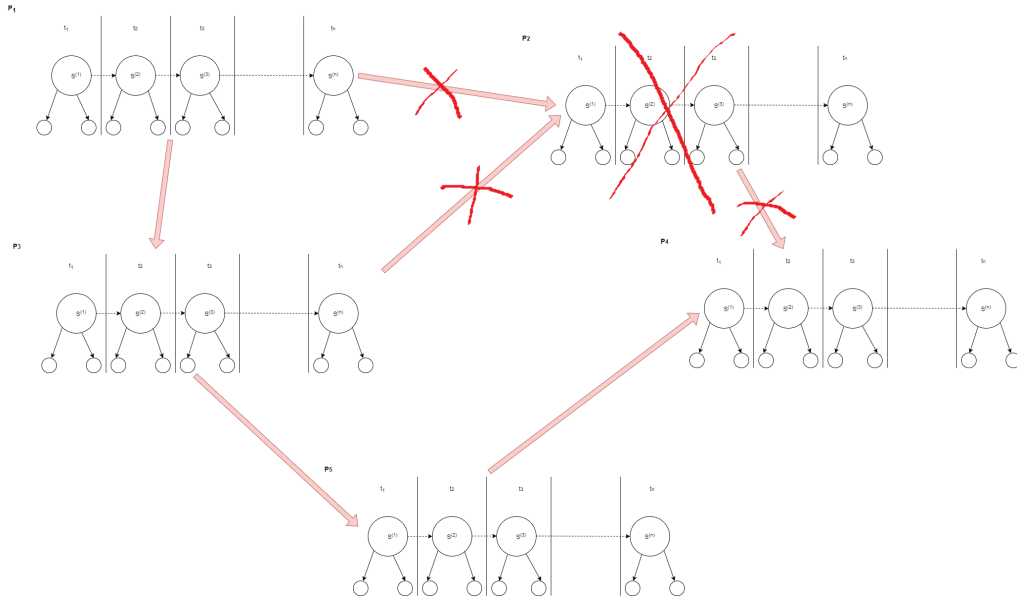


Figure 5.1: An influence structure between stochastic processes with a reduced graph structure for a temporal model set up.

Investigating advanced algorithms such as genetic algorithms or simulated annealing could enhance the robustness of structure learning. Lastly, conducting pilot implementations in collaboration with banks could validate the practical applicability of the proposed models and refine them based on real-world feedback.

This dissertation contributes to a novel perspective on modeling the influence of transaction volumes between ATMs using Bayesian learning. Through understanding and predicting the impact of ATM deactivation, financial institutions can enhance their operational strategies and improve reliability. The methodologies and findings presented a foundation for further research and practical advancements in the field of probabilistic graphical modeling.

References

- [Ajoodha and Rosman 2017] Ritesh Ajoodha and Benjamin Rosman. Tracking influence between naïve bayes models using score-based structure learning. pages 122–127, 11 2017.
- [Ajoodha and Rosman 2020] Ritesh Ajoodha and Benjamin Rosman. Discovery of influence between processes represented by hidden markov models. In *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pages 1–7, 2020.
- [Ajoodha and Rosman 2021] Ritesh Ajoodha and Benjamin Rosman. Using score-based structure learning to computationally learn direct influence between hierarchical dynamic bayesian networks. In *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, pages 1–6, 2021.
- [Ajoodha 2018] R. Ajoodha. Influence modelling and learning between dynamic bayesian networks using score-based structure learning. 2018.
- [Andrawis *et al.* 2011] Robert Andrawis, Amir Atiya, and Hisham El-Shishiny. Forecast combinations of computational intelligence and linear models for the nn5 time series forecasting competition. volume 27, 2011.
- [Arabani and Komleh 2018] Soodabeh Arabani and Hosein Komleh. The improvement of forecasting atms cash demand of iran banking network using convolutional neural network. volume 44, pages 3733–3743, 12 2018.
- [Armenise *et al.* 2012] R. Armenise, C. Birtolo, E. Sangianantoni, and L. Troiano. Optimizing atm cash management by genetic algorithms. *International Journal of Computer Information Systems and Industrial Management Applications.*, 2012.
- [Bandara 2017] K. Bandara. Forecasting across time series databases using long short-term memory networks on groups of similar series. 2017.
- [Bati and Gozupek 2017] Seyma Bati and Didem Gozupek. Joint optimization of cash management and routing for new-generation automated teller machine networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, PP:1–15, 08 2017.
- [Bhandari and Gill 2016] Renu Bhandari and Jasmeen Gill. An artificial intelligence atm forecasting system for hybrid neural networks. volume 133, 1 2016.
- [Biedermann and Taroni 2023] Alex Biedermann and Franco Taroni. *Bayesian Networks and Influence Diagrams*, pages 271–280. 01 2023.
- [Bilir and Doseyen 2018] C. Bilir and A. Doseyen. Optimization of atm and branch cash operations using an integrated cash requirement forecasting and cash optimization model. *Business and Management Studies: An International Journal*, 6:237–255, 2018.
- [Bishop 2006] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

- [Broda *et al.* 2014] Piotr Broda, Levajkovi Tijana, Kresoja Milena, Marceta Marina, Mena Hermann, Nikoli Mladen, and Stojancevic Tijana. Optimization of atm filling-in with cash. *Mathematics in Industry*, 3 2014.
- [Chen *et al.* 2020] Xianjiang Chen, Huiru Zheng, Haiying Wang, and Tianhai Yan. Bayesian network approach to modeling nitrogen utilization efficiency of dairy cows. pages 2730–2736, 12 2020.
- [Chotayakul *et al.* 2013] Supatchaya Chotayakul, Peerayuth Charnsethikul, Juta Pichitlamken, and John Kobza. An optimization-based heuristic for a capacitated lot-sizing model in an automated teller machines network. *Journal of Mathematics and Statistics*, 9:283–288, 10 2013.
- [Cisternas-Vera *et al.* 2016] F.O. Cisternas-Vera, A.L Montgomery, and W. van Hoeve. Reshaping bank branch networks due to mobile banking. *Semantic Scholar*, 2016.
- [Coyle *et al.* 2010] D Coyle, G Prasad, and M McGinnity. On utilizing self-organizing fuzzy neural networks for financial forecasts in the nn5 forecasting competition. 6 2010.
- [Dilijonas *et al.*] Darius Dilijonas, Sakalauskas Virgilijus, Kriksciuniene Dalia, and Simutis Rimvydas. Intelligent systems for retail banking optimization. *Semantic Scholar*.
- [du Toit 2011] J. du Toit. Atm cash management for a south african retail bank. 2011.
- [Ekinci *et al.* 2014] Yeliz Ekinci, Jye-Chyi Lu, and Ekrem Duman. Optimization of atm cash replenishment with group-demand forecasts. *Expert Systems with Applications*, 12 2014.
- [Fallahfati *et al.* 2022] Alireza Fallahfati, Mohammadreza Aghaaminiha, Sara Akbarghanadian, and Gary Weckman. Forecasting atm cash demand before and during the covid-19 pandemic using an extensive evaluation of statistical and machine learning models. volume 22, page 164, 2 2022.
- [Genevois *et al.* 2015] M.E Genevois, D. Celik, and H. Z. Ulukan. Atm location problem and cash management in automated teller machines. *International Journal of Industrial and Manufacturing Engineering*, 9(7), 2015.
- [Ghodrati *et al.* 2013] Ahmadreza Ghodrati, Hassan Abyak, and Abdorreza Sharifhosseini. Atm cash management using genetic algorithm. *Management Science Letters*, 3:2007–2014, 07 2013.
- [Gurgul and Suder 2016] Henryk Gurgul and Marcin Suder. Calendar and seasonal effects on the size of withdrawals from atms managed by euronet. *Statistics in Transition. New Series*, 17:691–722, 12 2016.
- [Heckerman 1995] David Heckerman. *A Tutorial on Learning With Bayesian Networks*. Technical Report MSR-TR-95-06, March 1995.
- [Hossain *et al.* 2019] Niamat Ullah Ibne Hossain, Raed Jaradat, Seyedmohsen Hosseini, Mohammad Marufuzzaman, and Randy K. Buchanan. A framework for modeling and assessing system resilience using a bayesian network: A case study of an interdependent electrical infrastructure system. *International Journal of Critical Infrastructure Protection*, 25:62–83, 2019.
- [Hosseini *et al.* 2016] Seyedmohsen Hosseini, Abdullah Khaled, and Md Sarder. A general framework for assessing system resilience using bayesian networks: A case study of sulfuric acid manufacturer. *Journal of Manufacturing Systems*, 41, 10 2016.
- [Jaradat *et al.* 2020] Raed Jaradat, Niamat Ullah Ibne Hossain, Safae el Amrani, Mohammad Marufuzzaman, Randy Buchanan, Christina Rinaudo, and Michael Hamilton. Modelling and assessing interdependencies between critical infrastructures using bayesian network: A case study of inland

- waterway port and surrounding supply chain network. *Reliability Engineering System Safety*, 198:106898, 02 2020.
- [Jenkins 1970] G.M. Jenkins. Time series analysis forecasting and control. 1970.
- [Jörg 2011] Wichard Jörg. Forecasting the nn5 time series with hybrid models. pages 700–707, 5 2011.
- [Kamini *et al.* 2014] V. Kamini, V. Ravi, and D. N. Kuma. Chaotic time series analysis with neural networks to forecast cash demand in atms. *2014 IEEE International Conference on Computational Intelligence and Computing Research*, 2014.
- [Kammouh *et al.* 2020] Omar Kammouh, Paolo Gardoni, and G. Cimellaro. Probabilistic framework to evaluate the resilience of engineering systems using bayesian and dynamic bayesian networks. *Reliability Engineering [?] System Safety*, 198:106813, 06 2020.
- [Keyter 2004] M. Keyter. The application of regression analysis to forecast transaction volumes on atm devices. 2004.
- [Kiyaei and Kiaee 2021] Mohammadhossein Kiyaei and Farkhondeh Kiaee. Optimal atm cash replenishment planning in a smart city using deep q-network. In *2021 26th International Computer Conference, Computer Society of Iran (CSICC)*, pages 1–5, 2021.
- [Koller and Friedman 2009] Daphne Koller and Nir Friedman. Probabilistic graphical models: principles and techniques. *MIT press*, 2009.
- [Kurdel and Sebestyenova 2013] P. Kurdel and Jolana Sebestyenova. Routing optimization for atm cash replenishment. *International Journal of Computers*, 7:135–144, 01 2013.
- [Lazaro *et al.* 2017] Jorge Lazaro, Alvaro Jimenez, and Akiko Takeda. Improving cash logistics in bank branches by coupling machine learning and robust optimization. *Expert Systems with Applications*, 9 2017.
- [Leng and Prasad 2004] Geng Leng and McGinnity Thomas Prasad, Girijesh. An on-line algorithm for creating self-organizing fuzzy neural networks. 7 2004.
- [Li *et al.* 2009] Yaoguo Li, Huiye Sun, Chen Zhang, and Li Guohui. Sites selection of atms based on particle swarm optimization. volume 2, pages 526 – 530, 08 2009.
- [Li *et al.* 2020] K. Li, T. Zhang, and R. Wang. Deep reinforcement learning for multi-objective optimization. *Journal of IEE Transactions on Cybernetics*, 2020.
- [Liu *et al.* 2020] Ming Liu, Zhongzheng Liu, Feng Chu, Feifeng Zheng, and Chengbin Chu. A new robust dynamic bayesian network approach for disruption risk assessment under the supply chain ripple effect. *International Journal of Production Research*, 59, 11 2020.
- [Matejová *et al.* 2023] Miroslava Matejová, Oliver Lohaj, Zuzana Pella, and Jan Paralic. Modeling the influence of factors associated with atherosclerosis. pages 000063–000068, 01 2023.
- [Mohammadi and Wit 2014] Reza Mohammadi and E. Wit. Bayesian structure learning in sparse gaussian graphical models. *arXiv*, arXiv:1210.5371v6:28, 05 2014.
- [Mubiru and Nalubowa Ssempijja 2024] Kizito Mubiru and Maureen Nalubowa Ssempijja. Modelling and optimization of atm cash-loading under stochastic demand. *Discover Analytics*, 2, 09 2024.
- [Nigmatuli and Chaganva 2022] G.A Nigmatuli and O.B Chaganva. Research of an optimization model for servicing a network of atms and information payment terminals. *arXiv*, 10 2022.

- [Pearl 1988] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.
- [Rafi *et al.* 2020] M. Rafi, M. Wahab, M Khan, and H. Raza. Atm cash prediction using time series approach. *2020 3rd International Conference on Computing, Mathematics and Engineering Technologies*, 2020.
- [Rajwani *et al.* 2018] A. Rajwani, T. Syed, B. Khan, and S. Behlim. Regression analysis for atm cash flow prediction. *2017 International Conference on Frontiers of Information Technology (FIT)*, 2018.
- [Ramírez and Acuña 2011] Cristián Ramírez and Gonzalo Acuña. Forecasting cash demand in atm using neural networks and least square support vector machine. In César San Martin and Sang-Woon Kim, editors, *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 515–522, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
- [Rodriguez and Lozano 2008] J.D. Rodriguez and J.A. Lozano. Multi-objective learning of multi-dimensional bayesian classifiers. *Eighth International Conference on Hybrid Intelligent Systems*, pages 501–506, 2008.
- [Sarveswararao *et al.* 2022] Vangala Sarveswararao, Vadlamini Ravi, and Yelleti Vivek. Atm cash demand forecasting in an indian bank with chaos and hybrid deep learning networks. *Expert Systems with Application*, 211:118645, 8 2022.
- [Serengil and Ozpinar 2019] Sefik Serengil and Alper Ozpinar. Atm cash flow prediction and replenishment optimization with ann. 11:402–408, 01 2019.
- [Shachter 2007] Ross Shachter. *Model Building with Belief Networks and Influence Diagrams*, pages 177–201. 01 2007.
- [Simutus *et al.* 2007a] R. Simutus, D. Dilijonas, L. Bastina, J. Friman, and P. Drobinov. Optimization of cash management for atm network. *Information Technology and Control*, 36, 2007.
- [Simutus *et al.* 2007b] Rimvydas Simutus, Darius Dilijona, Lidija Bastina, and Josif Friman. A flexible neural network for atm cash demand forecasting. *Conference on Computational Intelligence, Man-Machine Systems and Cybernetics*, 12 2007.
- [Suder *et al.* 2007] Marcin Suder, Henryk Gurgul, Belem Barbosa, Artur Machno, and Lukasz Lach. A flexible neural network for atm cash demand forecasting. *Conference on Computational Intelligence, Man-Machine Systems and Cybernetics*, 12 2007.
- [Tang *et al.* 2020] Junqing Tang, Hans Heinimann, Ke Han, Hanbin Luo, and Botao Zhong. Evaluating resilience in urban transportation systems for sustainability: A systems-based bayesian network model. *Transportation Research Part C: Emerging Technologies*, 121:102840, 12 2020.
- [Thanh *et al.* 2023] Bui Tien Thanh, Van Tuan Dinh, Anh Chi Tuan, Van Dai1 Nguyen, Dinh Nguyen Tai Quang1, Thu Thuy Nguyen, and Thi Xuan Hoa Nguye. Multiobjective logistics optimization for automated atm cash replenishment process. *Intelligence of Things: Technologies and Applications*, 9 2023.
- [Van Anholt and Vis 2010] Roel Van Anholt and I.F.A. Vis. An integrative online atm forecasting and replenishment model with a service level constraint. 01 2010.
- [Venkatesh *et al.* 2013] Kamini Venkatesh, Vadlamini Ravi, Anita Prinzie, and Dirk Van den Poel. Cash demand forecasting in atm by clustering and neural networks. *European Journal of Operational Research*, 232:383–392, 8 2013.

- [Wainwright and Jordan 2008] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1–2):1–305, jan 2008.
- [Zaidi *et al.* 2010] Abbas Zaidi, Faisal Mansoor, and P. Papantoni-Kazakos. Theory of influence networks. *Journal of Intelligent and Robotic Systems*, 60:457–491, 12 2010.
- [Zheng *et al.* 2016] Huiru Zheng, Haiying Wang, and Tianhai Yan. Modelling enteric methane emissions from milking dairy cows with bayesian networks. pages 1635–1640, 12 2016.