6-29-2021

Interpretability of AI in Computer Systems and Public Policy

Farzana Beente Yusuf
Florida International University, fyusu003@fiu.edu

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

INTERPRETABILITY OF AI IN COMPUTER SYSTEMS AND PUBLIC POLICY

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Farzana Beente Yusuf

2021

To: Dean John L. Volakis
College of Engineering and Computing

This dissertation, written by Farzana Beente Yusuf, and entitled Interpretability of AI in Computer Systems and Public Policy, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Raju Rangaswami

Mark Finlayson

Mitsunori Ogihara

Malek Adjouadi

Giri Narasimhan, Major Professor

Date of Defense: June 29, 2021

The dissertation of Farzana Beente Yusuf is approved.

Dean John L. Volakis
College of Engineering and Computing

Andrés G. Gil
Vice President for Research and Economic Development
Dean of the University Graduate School

Florida International University, 2021

DEDICATION

I dedicate this dissertation work to my precious family, especially my beloved parents (Kamrun Nahar and Md. Yusuf Ali). It would have been impossible to accomplish this without their continuous support, love, patience, and understanding.

ACKNOWLEDGMENTS

I want to express my gratitude to my advisor, Dr. Giri Narasimhan; It would be challenging to complete my Ph.D. journey without his constant support and encouragement. He taught me to be persistent in the face of adversity and overcome any hurdles. His unwavering support has enabled me to excel in my Ph.D. program and produce numerous outstanding research papers. I am grateful for his tireless efforts in preparing me to tackle any problem either academically or in real life. I want to thank my committee members, Dr. Mark Finlayson, Dr. Mitsunori Ogihara, Dr. Raju Rangaswami, Dr. Malek Adjouadi, for their time, encouragement, and support. Also, many thanks to Dr. Jason Liu and Dr. Miguel Alonso for their valuable advice and guidance in the process. Finally, I am indebted to Dr. Sukumar Ganapati and Dr. Shaoming Cheng for their valuable feedback, support, and suggestions to succeed in my journey. I can not thank Dr. Sam Ganzfried enough for mentoring me during the early years of my Ph.D. journey. I appreciate his support and guidance through the difficult times. I appreciate the support of my coworkers and colleagues from the BioRG lab at FIU, especially Daniel Ruiz-Perez, Vitalii Stebliankin, Camilo Valdes, Musfiqur Rahman Sazal, Arpit Mehta, for everything they have done for me. For the collaboration and helping with many parts of the thesis, I am indebted to my mentees, Eysler Paz and Rukmangadh Sai Myana. I appreciate the help from Liana V. Rodriguez and Steven Lyons from SyLab for their help throughout the collaboration.

I am grateful to my parents for supporting me in chasing my dream and always being by my side no matter what. From the bottom of my heart, I would like to express my gratitude to both of them. My mother, Kamrun Nahar, the strongest person I have ever seen in my life, gives me priceless and infinite love and support, which helps me to overcome any obstacles and struggles. My father, Md. Yusuf Ali loves me unconditionally and supports me in any situation, which helps me thrive each day. Their overwhelming

love helps me to keep going. I also like to thank my siblings, Shahadat Hossain Khan, Sakhawat Hossain Khan, Al Amin Dhurbo, Farhana Binte Yousuf, for being my friend, mentor, and support throughout my life. I am grateful to my brother-in-law and sisters-in-law for being a part of this journey. A special thanks to all my nephews and nieces, Nihan, Hamim, Samin, Mushfira, Ayesha. Last but not least, I owe my gratitude to my best friends, Sanjari Ahmed Orko, Labiba Jahan, Masudur R. Siddiquee, who are not any less than a family. I want to express my gratitude to my friends in the USA, who added colors to my life during this remarkable journey. Many thanks to Nusrat Yeasmin, Shagufta Gaffar, Tahnin Tariq Tani, Naznin Akter, Saki Rezwana, Maliha Tabassum Huma, Proyash Podder, Prianka Bhowal, Akash Bishwas, Nusrat Jahan, and many more.

ABSTRACT OF THE DISSERTATION

INTERPRETABILITY OF AI IN COMPUTER SYSTEMS AND PUBLIC POLICY

by

Farzana Beente Yusuf

Florida International University, 2021

Miami, Florida

Professor Giri Narasimhan, Major Professor

Advances in Artificial Intelligence (AI) have led to spectacular innovations and sophisticated systems for tasks that were thought to be capable only by humans. Examples include playing chess and Go, face and voice recognition, driving of vehicles, and more. In recent years, the impact of AI has moved beyond offering mere predictive models into building interpretable models that appeal to human logic and intuition because they ensure transparency and simplicity and can be used to make meaningful decisions in real-world applications. A second trend in AI is characterized by important advancements in the realm of causal reasoning. Identifying causal relationships is an important aspect of scientific endeavor in a variety of fields. Causal models and Bayesian inference can help us gain better domain-specific insight and make better data-driven decisions because of their interpretability.

The main objective of this dissertation was to adapt theoretically sound AI-based interpretable data-analytic approaches to solve domain-specific problems in the two unrelated fields of Storage Systems and Public Policy. For the first task, we considered the well-studied problem of cache replacement problem in computing systems, which can be modeled as a variant of the well-known Multi-Armed Bandit (MAB) problem with delayed feedback and decaying costs, and developed an algorithm called EXP4-DFDC. We proved theoretically that EXP4-DFDC exhibits an important feature called vanishing regret. Based on the theoretical analysis, we designed a machine learning algorithm called ALeCaR, with

adaptive hyperparameters. We used extensive experiments on a wide range of workloads to show that ALeCaR performed better than LeCaR, the best machine learning algorithm for cache replacement at that time. We concluded that reinforcement machine learning can offer an outstanding approach for implementing cache management policies.

For the second task, we used Bayesian networks to analyze the service request data from three 311 centers providing non-emergency services in the cities of Miami-Dade, New York City, and San Francisco. Using a causal inference approach, this study investigated the presence of inequities in the quality of the 311 service to neighborhoods with varying demographics and socioeconomic status. We concluded that the services provided by the local governments showed no detectable biases on the basis of race, ethnicity, or socioeconomic status.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Over the last decade, the field of *Artificial Intelligence* (AI), powered by *Machine Learning* (ML) and *Deep Learning* (DL), has transformed the world we live in with far-reaching applications like self-driving cars [Bojarski et al., 2016, Krizhevsky et al., 2012], strategy development in games [Mnih et al., 2013, Ganzfried and Yusuf, 2017], medical diagnostics [Chen et al., 2017, Beam and Kohane, 2018], natural language processing [Sundermeyer et al., 2012, Graves et al., 2013, Chung et al., 2014]. ML and DL, sub-fields of AI, enable a system to learn from experience and improve its performance over time in an autonomous fashion and are extensively applied to solve many domain-specific problems, where they have been used as a black box mainly for prediction or forecasting.

ML models are designed to be predictive and excel in connecting the input data to the output and finding patterns from them. However, they fall short in reasoning about cause-effect relationships. Just as with ML and DL, a second trend in AI is characterized by major advances in the field of causal reasoning. Identifying causal relationships and applying causal reasoning is an integral part of scientific inquiry, and is applicable to a wide range of domains including life sciences, epidemiology, environment, economics, finance, behavior, public policy, and much more. Understanding cause-and-effect relationships can help us make better data-driven decisions. Traditionally, causality is studied by performing controlled experiments, typically in the form of randomized trials. However, Pearl and others [Pearl, 2009, Pearl and Mackenzie, 2018] have developed computational approaches to perform causal inferencing. Causal inferencing is a collection of AI techniques that allow us to infer causal relationships using observational data. Once a reliable causal

model is built, causal inferencing can also help us to infer the effect of interventions and counterfactuals, thus allowing us to answer "what if ..." questions for the modeled systems.

A third important trend in AI is the effort to build more "generalizable" AI models that can explain the decisions made by the model. For example, if an automatic loan approval system rejects an applicant's loan, or if a self-driving car is involved in an accident, then it is natural to demand an explanation. Model interpretability allows us to understand an outcome or a prediction, not just be informed of the outcome, thus allowing us to treat the algorithm as something beyond a "black box" that predicts outputs from inputs. To tackle the issue of interpretability, a new field of AI has emerged known as *Explainable AI* (XAI) and has been applied extensively to making neural network models more explainable [Goebel et al., 2018, Holzinger, 2018, Hagras, 2018, Samek et al., 2019]. Model interpretation comprehends and explains the what, why, and how of the decision-making process. Transparency, the ability to query, and the ease with which humans can grasp model decisions are essential in model interpretation.

It is also important to ensure that AI models are unbiased. There are numerous real-world circumstances, i.e., predicting potential criminals, judicial sentence risk scores, credit scoring, fraud detection, health assessment, self-driving, in which biased models can have disastrous consequences. In many ML models, bias is eliminated by careful choice of training sets. Note that it may be possible to measure the extent of bias if we have an interpretable ML model, but an unbiased model would not make it interpretable. In causal models, the extent of bias may be more readily quantifiable.

The main objective of this proposal was to develop theoretically sound AI-based interpretable data-analytic approaches to solve domain-specific problems in *Storage Systems* and *Public Policy*. The research reported in this dissertation also focused on achieving domain-specific insight, thus allowing us to better understand and interpret the system, and find better ways to make the methods adaptive and automatic. To achieve this, the

methods were applied to data from two vastly different fields, i.e., Storage Systems and Public Policy.

Our first application area was cache management. Every cache management process handles billions of memory requests and make decisions about which memory items to retain in the cache at any given time. It is natural to wonder if it is possible to learn from prior mistakes and to improve a cache replacement policy over time. Although cache management has improved over the last few decades, ML has only been applied with limited success [Ari et al., 2002, Liu et al., 2019]. Improved predictions of what to retain in the precious small cache can dramatically impact system efficiency. However, this requires knowing the future. A natural question to ask is whether predictive and interpretable ML models can be utilized to outperform the existing non-ML algorithms. A recent algorithm designed at FIU called LeCaR answered that question in the affirmative. An in-depth study was needed to understand why the ML-based approach used by LeCaR worked well and whether we can explicitly identify what the algorithm learned by interpreting the models built. These questions are particularly relevant today in the context of different ML models [Goodfellow et al., 2016, Amvrosiadis et al., 2018] where interpretation of the models has become a challenge. The answers have far-reaching implications. A slight improvement of a cache replacement algorithm could mean energy savings on every computing device that uses it. The first work proposed here aims to understand the effectiveness of ML approaches applied to cache management, both from a theoretical and a practical perspective. Therefore, the first application of ML-based system improvement reported in this dissertation focuses on applying interpretable ML to the storage subsystem of a computing system with the goal of improving its performance.

Our second application area was in the field of public policy. Traditional data science methods for analyzing big data have mainly relied on correlations (e.g., regression, linear models, etc.). Emerging methods based on Pearl's causal inferencing techniques provide

new avenues for understanding causal relationships between variables. Causal models capture more than correlations and associations, allowing us to extract meaningful relationships, and making it possible to analyze the effects of interventions. Understanding the causes behind the effects is critical in decision-making processes. Additionally, if one intends to act on a forecast or decide to intervene to change the existing policies, then causal inferencing becomes even more important. The process of making public policy decisions made using administrative data was explored in this dissertation. By using conditional probabilities, Bayesian networks provide an approach to distinguish between direct and indirect dependencies and to identify common effects from the observational data. We applied causal Bayesian networks to publicly available administrative data from 311 call centers, which provide non-emergency services to residents. Our secondary goal was to use causal inferencing to study equity issues in the 311 centers in different cities of USA, i.e., Miami-Dade County, New York city, and San Francisco city. In the process, this project also aimed to identify the challenges of applying causal inference on public policy domain data and provide recommendations for the use of causal inferencing using interpretable causal BNs.

1.2 Research Components

As mentioned earlier, the main objective of this proposal was to develop data-analytic approaches to improve decision making in *Storage Systems* and *Public Policy*. This is described further in the two components mentioned below.

1. **Component 1– Application of ML on Cache replacement problem:** The cache, a mainstay in nearly every computing device, is the fastest storage device for a program to access. When a data item is requested, it is first searched in the cache. If found, the cache hit results in the fastest access. Else, a cache miss occurs, and

the item is brought into the cache for further processing from a storage device that is a lot slower (e.g., main memory or secondary storage). Caches are expensive, and the cost rises steeply with size and speed of access. Since the small cache is almost always full, some item is evicted from it on every miss to make room for the new data. The evicted item is chosen using the *cache replacement policy*. If the cache is provisioned to be sufficiently correctly and if cache replacement is managed well, then it is equivalent to having the entire memory needs of the program stored in the fastest storage device. Thus the efficiency of the system can be significantly enhanced if the cache replacement policy is successful. However, its success requires knowing the future. The basic algorithms of Least Recently Used (LRU), Least Frequently Used (LFU), and CLOCK [Corbato, 1968] have identified key features (recency and frequency of items) to implement an efficient replacement policy. Algorithms such as LIRS [Jiang and Zhang, 2002], ARC [Megiddo and Modha, 2003, 2004] appear to anticipate the future better than the aforementioned basic algorithms in many settings and predates many innovations in ML. LeCaR represents a recent breakthrough in cache replacement policies and is a ML-based "online reinforcement learning" algorithm [Vietri et al., 2018].

(a) **Theoretical Analysis of LeCaR:** LeCaR [Vietri et al., 2018], an online adaptive cache replacement algorithm, demonstrated outstanding empirical performance over state-of-the art cache replacement algorithm. The LECAR algorithm makes a simple, but elegant, assumption that on any cache miss, it is sufficient for the algorithm to pick from one of only two policies: Least Recently Used (LRU) and Least Frequently Used (LFU). In this dissertation, we analyzed LeCaR theoretically with the goal of gaining insights into the worst-case performance of the algorithm.

(b) **Designing improved versions of LeCaR:** Understanding the impact of the hyperparameters in LeCaR is essential as they are an integral part of adaptation. Updating the regret values at each step and the fixed design choices might result in performance bottleneck of the algorithm since they do not incorporate systems dynamics. An improved version that assigns theoretically optimal values for some of the hyperparameters and automates the setting of others was designed and implemented.

2. **Component 2– Causality on public policy domain:** The second aim of this dissertation was to illustrate the application of causal inference method to administrative data. The task was divided into two chapters.

   (a) **Causal Inferencing and its Challenges: The Case of 311 Data** We applied causal Bayesian networks method to 311 data from Miami-Dade County, Florida (USA). The 311 centers provide non-emergency services to residents. The 311 data are large and granular. We explored the equity issues and biases that might exist in this particular type of service requests. As a case study, the relationship between population characteristics (independent variables), request volume, and completion time (dependent variables) was examined to identify any racial or ethnic disparities that may be present in the observational data.

   (b) **Comparative analysis of 311 data for different cities** We looked at the limits of the 311 data analytics: Processes, Potential Benefits, and Limitations of the Bayesian Network Approach to causal inference when applied to admisitrative dataset. We provided an improved framework to overcome the limitations. To compare how effective the method was in terms of completion time, we looked at data from three different cities: Miami-Dade County, New York City,

and San Francisco. The causal (signed) Bayesian Network models that were developed suggested that demographic and socioeconomic characteristics had little or no influence on the target variables, such as completion time and call request volumes. The data did not support the existence of demographic or socioeconomic bias in the provision of non-emergency services to residents of these cities according to our findings.

1.3 Contributions

1.3.1 Theoretical analysis of LeCaR

We studied a variant of the well-known *Multi-Armed Bandit* (MAB) problem with delayed feedback, costs decaying with delay, and costs vanishing after a threshold. Previously studied variants focused on the case where costs increased with delays, and are hence not applicable to the variant under study. We presented an algorithm called EXP4-DFDC for the delayed-feedback-decaying-cost variant. Theorem 2.4.2 shows that it is guaranteed to have vanishing regret. Cumulative regret for the EXP4-DFDC algorithm is of the form O($\sqrt{KT \ln N}$). As a result, average regret vanishes with increasing time, since $R_{\text{EXP4-DFDC}}(t)/T = O(T^{-1/2}) \to 0$, as $T \to \infty$.

Theorem 1.3.1 *Given $K, T > 0$, learning rate $\eta \in (0, 1]$, a family of N experts, and an assignment of arbitrary rewards regularized by a decaying cost that depends on delay d, the expected regret of the EXP4-DFDC algorithm can be upper bounded by $R_{\text{EXP4-DFDC}}(T)] \leq 2\eta T + (K \ln N)/\eta$.*

A corollary of Theorem 1.3.1 is that LeCaR, a learning algorithm based on EXP4-DFDC, with any fixed non-negative learning rate, has a regret value that vanishes with increasing time horizon.

Improvement of LeCaR based on theoretical analysis The optimal learning rate at which regret is minimized can be derived theoretically (as a function of time) from Theorem 2.4.2. We have designed an adaptive version of LeCaR, called OLECAR, with learning rate set as recommended by the theoretical derivation.

1.3.2 Adaptive LeCaR: ALeCaR

Inspired by the gradient-based approach [Robbins and Monro, 1951], we proposed a novel approach to make the learning rate (η) adaptive. ALeCaR is a gradient-based variant of LeCaR that has been shown to improve the performance over state-of-the art algorithms. ALeCaR outperforms the best-known algorithms (ARC, LIRS, and DLIRS) 26% of the times, is competitive 48% of the times. ALeCaR paved the way for the development of CACHEUS [Rodriguez et al., 2021], the best caching algorithm to date and enabled the development of a framework for combining cutting-edge caching techniques like ARC or LIRS with a supplementary expert like LFU to handle a broad range of workloads efficiently and effectively.

1.3.3 Causal Inference applied to the 311 Data

We applied the causal inference method to administrative data. First, we used Bayesian networks to analyze 311 data from Miami-Dade County, Florida (USA). Residents use the 311 centers for non-emergency services. The 311 data is extensive and detailed. However, due to missing or impure data, insufficiency, and latent confounders, inferring causation from administrative data poses a number of issues. Finally, we attempted to improve the inferencing technique by integrating prior domain knowledge into the Bayesian networks by adding temporal information, integrating blacklisting (list of edges in the blacklist never appear in the learned network), and extending it to new cities, i.e., New York City and San

Francisco, with more observations. The purpose of this study was to find out whether there are any equity concerns or biases in this type of service request. The relationship between demographic characteristics (independent factors) and request volume and completion time (dependent variables) was investigated as a case study in order to find any discrepancies from observational data. According to the findings, there are no biases in the services given to any demographic, socioeconomic, or geographic categories. The goal of this study was to use the causal inference method to analyze the 311 administrative data to learn more about how local governments respond to service requests and whether government service provision is consistent across demographic, socioeconomic, and geographic variables. More importantly, this research provided a framework for performing causal inference on 311 datasets that can be easily applied to others cities or regions. Finally, we discussed the difficulties and challenges of using the causal method with this type of dataset.

1.4 Outline

The dissertation is organized as follows. First, we introduced a new variant of the traditional MAB problem to map the cache replacement problem. We provided a theoretical analysis of the solution *delayed feedback and decaying cost* variant of the EXP4 algorithm named EXP4-DFDC. Finally, we apply the theoretical analysis to the LeCaR algorithm in the Chapter 2. Next, we described the adaptation of the learning rate of LeCaR along with detailed performance analysis in the Chapter 3. We discussed the application of *causal Bayesian networks* on the 311 datasets and provide insights, while taking on the challenges of interpreting the analysis in Chapter 4. Finally, we extended the causal framework to overcome the challenges and applied it to two more cities in Chapter 5 before providing concluding remarks in Chapter 6.

CHAPTER 2

CACHE REPLACEMENT AS A MAB WITH DELAYED FEEDBACK AND DECAYING COSTS

2.1 Motivation

Software programs can be speeded up using software or hardware caches. The *Cache Replacement* problem is a fundamental problem in computer science that dictates the performance of such caches. Classical cache replacement algorithms such as *Least Recently Used* (LRU) and *Least Frequently Used* (LFU) are known to perform well on average but leave a lot of room for improvement. The *Adaptive Replacement Cache* (ARC) algorithm showed better performance by exploiting both frequency and recency properties of data items, and quickly became the state-of-the-art method for cache replacement after its invention [Megiddo and Modha, 2003, 2004]. More recently, some algorithms have demonstrated comparable or even better performance than ARC. These include LIRS [Jiang and Zhang, 2002] and DLIRS [Li, 2018]. Among the recent crop of outstanding cache replacement algorithms is the LeCaR algorithm invented by Vietri et al. [2018], which uses online reinforcement learning, thus opening a promising new direction to improve cache replacement with the help of machine learning. In this study, we aim to address the cache replacement problem by formulating a new variant of the *Multi-Armed Bandit* (MAB) problem and providing theoretical insight into how to improve the LeCaR algorithm for cache replacement.

The original MAB [Robbins, 1985] problem can be formulated as a T-round game where a player has access to a panel of N experts, each following a specific strategy. In round t, the player consults the N experts (arms), each of whom recommends one of K possible actions, represented as a K-dimensional binary vector, $\xi_i(t)$ with only one occurrence of a 1. Thus $\xi_i(t) \in \{0,1\}^K$ with only one 1. The action in round t is j if and

only if $\xi_i^j(t) = 1$, where $\xi_i^j(t)$ is the j-th component of $\xi_i(t)$. We also note that the proof can be generalized to the case when the recommendations from each expert is represented by a probabilistic vector, i.e., $\xi_i^j(t) \leq 1$ and $\sum_{j=1}^{N} \xi_i(t) = 1$. Each action is associated with a reward or cost value, which is provided in the form of feedback to the player immediately following the action. The problem is to find the strategy that results in the highest gain. A successful algorithm for MAB must strike a balance between acquiring new knowledge about the input (i.e., "exploration") and optimizing decisions based on existing knowledge (i.e., "exploitation").

Cache replacement can be considered as a variant of the MAB where, for each page request $\sigma(t)$, expert i recommends an action $\xi_i(t)$ indicating the page in the cache to be evicted, if a replacement is needed. A feedback for an action to evict page X in round t is provided in round t' if the eviction of page X triggers a "miss" when X is requested for the first time at a later round, t'. Therefore, the feedback could be delayed by an indeterminate amount, and the magnitude of the associated cost is inversely proportional to the time passed since the action was taken. Note that because of the application being considered, it is convenient for us to use the term "cost" instead of "reward", although they are equivalent.

Even though variants of MAB with delayed feedback have been studied previously [Weinberger and Ordentlich, 2002, Agarwal and Duchi, 2011, Langford et al., 2009, Neu et al., 2010, Desautels et al., 2014, Dudik et al., 2011], they cannot be applied to the cache replacement problem. First, in all existing studies, cost increases with increased delay. In our case, with larger delay, the cost associated with the chosen action decreases. Second, the cost vanishes when the delay is larger than a specified threshold. In this dissertation, we formulated a new MAB variant - *MAB with delayed feedback and thresholded decaying cost* (MAB-DFDC).

To solve the newly formulated problem, we introduced the EXP4-DFDC algorithm and showed that it exhibits vanishing regret over an increasing time horizon. Finally, we presentd OLECAR – an enhanced LECAR version with a theoretically optimal adaptive learning rate derived from regret minimization applied to EXP4-DFDC.

2.2 Background

2.2.1 Variants of MAB

The classical *Multi-Armed Bandit* problem was initially formulated by Robbins [1952, 1985]. The motivating application was to find a balance between two competing goals of finding the best treatment (exploration) and treating the patients as soon as possible from the best known treatments (exploitation) so far in clinical trails for treatments. The formulation of MAB in terms of a T-round repeated game is as follows: For each play t over T rounds, the player selects one of the K actions, $\xi_i(t) \in \{0,1\}^K, i = 1, \ldots, n,$, where $\sum_{j=1}^{K} \xi_i^j(t) = 1$ and obtains a cost of $x_j(t)$ for this play. The cost for each action is associated with an unknown probability distribution. The goal is to find the the best strategy to follow by selecting actions that will minimize the total cumulative cost, thus minimize regret. As a result, many different algorithms have been proposed depending on the regret minimization technique, which involves playing a mix of exploration and exploitation strategies.

Lai and Robbins [1985] proved that the player's regret over T rounds can be made as small as $O(\log T)$ in a stochastic setting based on the assumption that the cost distribution over the actions are random. As $T \to \infty$, the average regret vanishes. Littlestone and Warmuth [1994] proposed a *weighted majority algorithm* with N experts which makes at most $c(\log|N|+m)$ mistakes on that play sequence, where c is a fixed constant and m is

the number of mistakes made by the best expert. The player chooses the action based on the majority voting, and the weights are updated based on the associated cost. Vovk [1995] provided similar bounds on the cumulative loss of the learning algorithm for the prediction problem, where it never exceeds $cL + a \log N$ such that a and c are fixed constants, N is the number of experts, and L is the cumulative loss incurred by the best expert in the pool. Variants of algorithms to solve a worst-case stochastic MAB problem in an online framework were developed [Freund and Schapire, 1995, 1999] based on the previous mentioned works.

In certain applications, the cost function cannot be modeled by a stationary distribution. As an example, in a communication network, to find the best route for transmitting packets from a fixed number of possible options requires sophisticated statistical assumptions on the associated costs of the routes. In other applications, it might be impossible to determine the appropriate distribution. Therefore, a *non-stochastic multi-armed bandit* (or a *adversarial bandit*) makes no assumption about the nature of the cost generation process [Auer et al., 2002]. Instead of a well-defined stochastic distribution, the adversary takes control over the costs generation process. The adversary may be *oblivious* (where the costs of all actions at all rounds are selected in advance) or *non-oblivious* (where the adversary changes the distribution based on the player strategy) [Auer et al., 1995].

The oblivious *adversarial* MAB problem can be solved using the *Exponential Explore-Exploit* (EXP3) algorithm [Auer et al., 2002], which is a variant of the "Hedge" algorithm proposed by Freund and Schapire [1995]. "Exponential weight algorithm for Exploration and Exploitation using Expert advice" (EXP4) [Auer et al., 2002] extends the EXP3 algorithm for multiple experts settings. The standard weighted majority algorithm is not effective in non-stochastic settings because experts with large weights could prevent actions with potentially smaller costs that are delivered later in time. Both the EXP3 and EXP4 algorithm has an exploration parameter, which controls the probability of choosing the

arm at random. It have been shown that both EXP3 and EXP4 player strategies guarantee a worst-case regret of $O(\sqrt{T})$, where T is the number of rounds played [Auer et al., 2002].

Many variants of the MAB problem and solution approaches have been developed inspired by different applications, i.e., online advertisement, network communication modeling, and news recommendation system based on the adversarial bandit work of Auer et al. [2002]. In some of these cases, the feedback is not instantaneous, instead delayed due to some constraints. For example, Weinberger and Ordentlich [2002] studied *MAB with delayed feedback* in the adversarial full information setting (the player receives feedback information related to all the actions instead of only the chosen action). They discussed an application of this to prefetch pages in computer memory systems and proved vanishing regret. Mesterharm [2005] explored another variant of the full information setting when side information is available. For example, in a social network, to recommend a new friend to a user, it is possible to exploit the mutual friend information by the recommendation system. For such a setting, the algorithm by Mesterharm [2007] exhibits increasing regret to average delay for the adversarial environment. They also studied the stochastic setting and provided a regret bound, which increases with maximum delay when feedback is delayed. Langford et al. [2009] proved similar bounds for a sufficiently slow learner and exploited parallelism to design an online learning strategy that best uses multi-core architecture technology to solve the problem. Neu et al. [2010] formulated a MAB with an oblivious adversary and showed a multiplicative regret for a fixed and known delay when there is no side information available. Dudik et al. [2011] and Desautels et al. [2014] presented online learning algorithms for delayed feedback, resulting in a regret bound with an additive term dependent on the fixed delay and maximum delay for no side information and side information case.

Beygelzimer et al. proposed a solution of contextual bandits when only partial feedback is available. The work of Agarwal and Duchi [2011] and Joulani et al. [2013] on online

stochastic MAB showed that regret increases linearly with delay respectively for full and partial information delayed feedback setting. The *blinded bandit* addresses another variant when the feedback is not provided on the rounds the player switches to a different action [Dekel et al., 2014].

One important application is the online routing network problem. During the route switching of the streaming transmission, it takes time to compute the new transmission rate. Cooperative bandit problems in multi-player settings have been studied by Awerbuch and Kleinberg [2008], Szorenyi et al. [2013] in a stochastic and adversarial setting, where the players cooperate to find the best strategy using dynamic random networks. Cesa-Bianchi et al. [2016] explored delay and cooperation in non-stochastic bandits with an application to communication networks. Cesa-Bianchi et al. [2016] proved a regret bound for adversarial delayed feedback in a cooperative multi-agent setting, where regret increases with delay. Joulani et al. [2016] showed regret bounds of $\sqrt{(d+T)\ln K}$, where d is the total delay in feedback experienced over the T rounds with K available actions for full-information settings.

2.3 Problem formulation: MAB with delayed feedback

In this section, we formulate a new *MAB* variant with *Delayed Feedback and Decaying Cost* (MAB-DFDC) that is applicable to the cache replacement problem. MAB-DFDC differs from the existing literature in terms of the formulation of delayed feedback. In the MAB problem, there is no delay in obtaining the feedback. In contrast, MAB-DFDC allows delays in feedback to be greater than zero. In previous studies that allowed delays in feedback, cost and/or regret was modeled as increasing with the delay. But, in our proposed formulation, regret decreases as delay increases. As inspired by the cache replacement problem, if an evicted item is requested immediately after its eviction, the regret is much

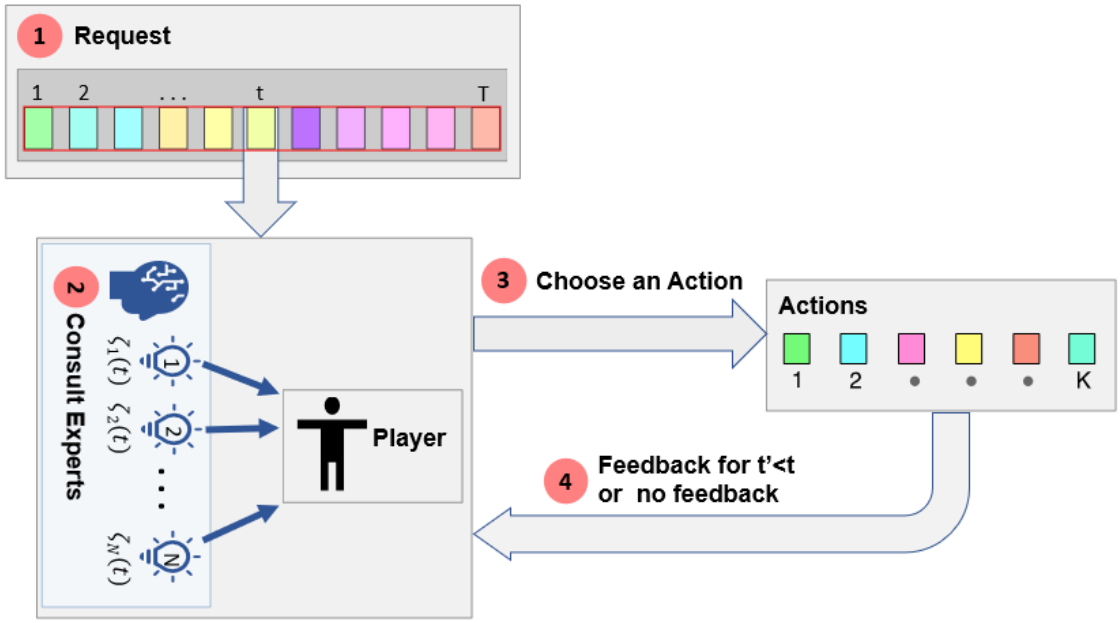


Figure 2.1: A schematic for the Multi-Armed Bandit problem with delayed feedback.

higher than if the request occurs after a considerable passage of time. Additionally, when the delay reaches a predefined threshold, our formulation ignores its contribution and the associated cost term vanishes from the regret bound equation. The intuition here is that once the regret due to an eviction becomes appreciably small, i.e., after a sufficient passage of time, it is not worth the bookkeeping effort to track its negligible impact and can be dropped altogether from further consideration.

We describe the delayed feedback, decaying cost version of MAB problem (MAB-DFDC), and borrow from the notation used for the adversarial MAB problem by Auer et al. [2002]. The problem is modeled as a game where the player's objective is to minimize costs. The player receives a sequence of requests and must respond with one of K actions at the end of each request. The player also has access to N experts who are consulted on every action. Every action has a cost and the player seeks to minimize the cost over T requests as T becomes large. As described earlier, the decaying cost is unknown until the action elicits a feedback, which may be delayed for an indeterminate amount of time.

However, the cost is a pre-specified function of the delay of the feedback – we assume a simple linear dependence. Fig. 2.1 provides a schematic and highlights the key concepts of the MAB-DFDC problem, from which the application to the cache replacement problem can be readily seen. The schematic describes 4 steps in each iteration, which are explained in detail below.

1. A request (e.g., a page request) is received for round t in a T-round game.
2. The player consults with N experts, each of whom recommends an action $\xi_i(t)$ as a (binary) vector of probabilities, where $\xi_i(t) = (p_1^i(t), \ldots, p_K^i(t))$ and $p_j^i(t) \in [0, 1], j = 1, \ldots, K$. As mentioned earlier, $\xi_i^j(t)$ can be thought of as the probability of choosing action j by expert i at time t. Note that all experts have access to the state of the system in making their expert recommendations, and that "no action" is a valid recommendation. In the cache replacement problem, an expert is a page replacement policy (e.g., LRU), and the action is the associated cache manipulation. "No action" is recommended if the item is found in cache (i.e., a hit). Each action is associated with a cost, which is revealed to the player after an indeterminate delay.
3. An action i_t is chosen by the player based on the recommendations of the experts and based on some pre-determined criteria. Therefore, $\xi_i^j(t) = 1$, if and only if $i_t = j$, and 0 otherwise. A "history" of actions is maintained in order to provide feedback when appropriate. History information may also be used by the experts to provide a recommendation. For the cache replacement scenario, a *hit* triggers "no action", but a *miss* results in an action that evicts a page resident in the cache to make room for the page requested in round t. Eviction information is stored in history to generate a feedback in a later round, if needed.
4. Finally, a feedback may be generated. The feedback may be for the current request or for a request from some previous round t', and its computation requires the

information stored in history. For the cache replacement problem, a hit triggers a "no action" and results in an immediate feedback with no cost. if a "miss" is encountered for the currently requested page, we search the page eviction history to find the round t' when the page was last evicted from the cache. If the currently requested page is found in history, we provide the feedback for the action from round t'. If the requested page is not found in history, no feedback is provided.

The classical MAB problem assumes that $t = t'$, i.e., that the feedback is immediate for every action. In our case, the delay, $d = t - t'$, is defined as the number of rounds between when the action is taken and when the feedback was received for that particular action. We assume that the cost is a function of the delay. Consequently, its computation may require an infinite history. Since this is impractical in real applications, we assume that the history is truncated and is of bounded size. In the cache replacement application, the history size is bounded by some memory constraint and we assume that beyond a certain delay, the real cost is negligible and can be ignored. In other words, we do not distinguish between a request that is appearing for the first time (and therefore triggers a miss) and a request that appears after an extremely long time well after the item was last evicted from the cache.

Since feedback may be *delayed*, the penalty/cost for an action is a decaying function of the delay d. Also, if the delay (in feedback for any action) is higher than some predefined threshold, then it is assumed to be infinite, thus making the feedback cost for that particular action to be negligible and ignored totally.

Many variants of MAB have been proposed based on the way the cost function is defined. We assume that in round t, the vector of values of the cost function for each action are denoted by $\mathbf{x}(t) = (x_1(t), \ldots, x_K(t))$, where $x_j(t) \in [0, 1]$. In an oblivious model, the cost function does not depend on the player's action in the previous rounds. In an adversarial model, the cost function may change based on the previous actions. In each round, no feedback is provided on the other $K - 1$ actions that were not chosen by the

player. The main objective of the player is to minimize the cost (or maximize the reward) over a T-round game. The cumulative cost after T rounds for the best strategy is given by

$$C_{best}(T) = \min_{1 \leq i \leq N} \sum_{t=1}^{T} \xi_i(t) \cdot \mathbf{x}(t),$$

which assumes that the best of the N expert recommendations are followed in each round. Also, the cumulative cost incurred by an algorithm A selecting action $a(t)$ in round t is given by

$$C_A(T) = \sum_{t=1}^{T} x_{i_t}(t).$$

To measure the performance of adaptive learning algorithms, the concept of *regret* was borrowed from the literature on the theory of games. For any algorithm, *regret* can be defined in terms of cumulative difference between the costs of the best strategy in each step and the algorithm in consideration. Thus, the regret $R_A(T)$ of any algorithm A after round T can be calculated from Eq. (2.1) below.

$$R_A(T) = C_{best}(T) - C_A(T). \tag{2.1}$$

The main objective of any adaptive learning algorithm is to minimize the total regret over time, while ensuring that it vanishes over a long time horizon.

2.4 Algorithms

We first describe the EXP4 algorithm designed by Auer et al. [2002] before describing the EXP4-DFDC algorithm analyzed in this chapter. EXP4-DFDC extends the EXP4 algorithm for the *Delayed Feedback and Decaying Cost* settings.

2.4.1 Existing approach: EXP4 algorithm

The "Exponential weight algorithm for Exploration and Exploitation using Expert advice" (EXP4) algorithm for the non-stochastic multi-armed bandit problem assumes the existence

of a fixed set of N strategies/experts. At each round t, all N experts are consulted. In each round, the player either follows the advice of one of the experts or explores by picking a random action. If no prior knowledge is available about the experts, EXP4 initializes by assigning equal weights. Thus, $w_i(1) = 1, i = 1, \ldots, N$. The probability, $p_j(t)$, of picking action $j = \{1, \ldots, K\}$ in round t is proportional to the sum of the weights of the experts that recommend action j. If more experts recommend an action, that action will have higher probability to be chosen by the algorithm. This is interpreted as the "exploitation" of the information accumulated by the learning algorithm so far. If the player chooses to explore with a random action, the algorithm will choose an action randomly from the available options. In other words, $p_j(t)$, the probability of the player taking action j in round t is given by Eq. (2.2) below, for $j = \{1, \ldots, K\}$.

$$p_j(t) = (1 - \eta) \sum_{i=1}^{N} \frac{w_i(t) \times \xi_i^j(t)}{W_t} + \frac{\eta}{K}, \tag{2.2}$$

where η is the learning rate, $w_i(1) = 1, i = 1, \ldots, N$, and $W_t = \sum_{i=1}^{N} w_i(t)$. The learning rate η controls the amount of exploration and exploitation at each round. If the learning rate is too high, the algorithm will explore more and exploit less and vise versa.

Whenever some feedback is made available, the algorithm updates the weights according to Eq. (2.3) below. In the EXP4 algorithm, the estimated cost vector is denoted by $\hat{\mathbf{x}}(t)$, where each element $\hat{x}_j(t)$ of the vector is set to $x_j(t)/p_j(t)$ for $j = 1 \ldots K$ and upper bounded by the maximum possible value of actual cost $x_j(t)$ as in Auer et al. [2002]. The estimated cost $\hat{x}_{i_t}$ of any action i_t ensures that the actions with low probabilities are adjusted when they get picked in a later round. Otherwise, the algorithm will keep ignoring an action because the cost was high in an earlier round. In a later round, when the cost associated with that action becomes lower, it is not reflected, and the action never gets chosen due to its low probability. Also, given a sequence of random choices $i_1, i_2, \ldots, i_{t-1}$ of the previous rounds, the expectations of estimated cost of any random

action i_t taken at round t is guaranteed to be equal to the actual cost of that action. Consequently, $\mathbb{E}\left[\hat{x}_{i_t}(t)|i_1,\ldots,i_{t-1}\right] = x_{i_t}(t)$ As in Auer et al. [2002], the weight update rule is as follows:

$$w_i(t+1) = w_i(t)\ \exp\left(-\frac{\eta\hat{\mathbf{x}}(t)\cdot\xi_i(t)}{K}\right), \qquad i = 1,\ldots,N. \tag{2.3}$$

The same steps are applied repeatedly for T steps, allowing us to bound the regret for EXP4 as outlined in Cesa-Bianchi and Lugosi [2006], Auer et al. [2002].

Theorem 2.4.1 *[Auer et al., 2002] For any $K, T > 0$, for any learning rate $\eta \in (0, 1]$, for any family of N experts (including an uniform expert), and for any assignments of arbitrary costs, the following holds,*

$$R_{EXP4}(T) \leq (e-1)\eta T + \frac{K\ln N}{\eta}. \tag{2.4}$$

Classical EXP4 cannot be used in the bounded and delayed feedback settings because the update rule cannot be applied to rounds where no feedback is available or the feedback is delayed beyond a defined limit. Dudik et al. [2011] presented an algorithm devising a modification of EXP4 for *MAB with stochastic delayed feedback*. Meanwhile Neu et al. [2010] showed a multiplicative regret for the adversarial bandit case without any additional information. The full information case was developed by Joulani et al. [2016], showing regret bounds of $\sqrt{(d+T)\ln K}$, where d is the total delay in feedback experienced over the T rounds. Also the online learning case under delayed feedback for stochastic bandit optimization was analyzed by Desautels et al. [2014] and resulted in a regret bound involving a multiplicative increase that is independent of the delay and an additive term depending on the maximum delay.

2.4.2 New approach: EXP4-DFDC Algorithm

We describe, EXP4-DFDC, a variant of EXP4, to solve MAB-DFDC, the MAB problem with delayed feedback and decaying cost. In EXP4-DFDC, when feedback is non-zero, the weights of the experts get updated. A feedback is generated at round t, triggered by an action, $i_{t'}$, which was taken back in round t'. As the delay in feedback, $t - t'$, increases, the relative cost of the chosen action decays. Since regret is proportional to cost, the weights of experts who chose the action $i_{t'}$ are decreased using the estimated cost value observed at round t. Estimated cost is calculated as:

$$\hat{\mathbf{x}}_{i_t}(t) = \frac{x_{i_t}(t)}{d \times p_{i_t}(t)}. \tag{2.5}$$

Hence, the estimated cost calculation, which incorporates the decaying cost with delayed feedback in EXP4-DFDC is the major difference from the existing literature in the proposed algorithm. Finally, we want to draw a conclusion on the performance of the generic EXP4-DFDC and specific OLECAR algorithm. Below we will prove a theorem for the delayed feedback and decaying case along the lines of Theorem 2.4.1.

Theorem 2.4.2 *For any $K, T > 0$, for any learning rate, $\eta \in (0, 1]$, for any family of N experts, and for any assignment of arbitrary costs decaying with delay d, the expected regret of the algorithm can be upper bounded by: $R_{\textit{EXP4-DFDC}}(T) \leq 2\eta T + \frac{K \ln N}{\eta}$.*

Proof: As with other reinforcement algorithms, we formalize the regret bound for EXP4-DFDC. For the sake of completeness, we provide here the complete steps following the derivation for EXP4 [Auer et al., 2002] regret bound and adapt it for our delayed and decaying cost setting.

Note that $W_t = w_1(t) + w_2(t) + \ldots + w_N(t)$. We consider the ratio of the sum of weights over all iterations.

$$\frac{W_{t+1}}{W_t} = \sum_{i=1}^{N} \frac{w_i(t+1)}{W_t} = \sum_{i=1}^{N} \frac{w_i(t)}{W_t} \exp\left(-\frac{\eta}{K}\hat{\mathbf{x}}(t) \cdot \xi_i(t)\right) \tag{2.6}$$

Algorithm 1: EXP4-DFDC

Input: Learning rate $\eta \in (0, 1]$, delay d, delay threshold for feedback m

begin

Set $w_i(1) = 1$, for $i = 1, \ldots, N$;

for $t = 1, 2, \ldots$ **do**

Obtain expert advice $\xi_1(t), \ldots, \xi_N(t)$;

Set $W_t = \sum_{i=1}^{N} w_i(t)$;

for $j = 1, \ldots, K$ **do**

$p_j(t) = (1 - \eta) \sum_{i=1}^{N} \frac{w_i(t) \times \xi_i^j(t)}{W_t} + \frac{\eta}{K}$;

end

Select action $i_t \in [1, K]$ using probability distribution $p_j(t)$;

for $j = 1, \ldots, K$ **do**

$$\hat{x}_j(t) = \begin{cases} \frac{x_j(t)}{d \times p_j(t)}, & \text{if } i_{t'} = j;\ 1 \leq d \leq m \\ 0, & \text{otherwise} \end{cases}$$

end

for $i = 1, \ldots, N$ **do**

$$w_i(t+1) = w_i(t) \exp\left(-\frac{\eta \hat{\mathbf{x}}(t) \cdot \xi_i(t)}{K}\right)$$

end

end

end

Replacing $\frac{w_i(t)}{W_t}$ by $q_i(t)$ and setting $y_i(t) = -\hat{\mathbf{x}}(t) \cdot \xi_i(t)$ in Eq. (2.6) results in the following equation:

$$\frac{W_{t+1}}{W_t} = \sum_{i=1}^{N} q_i(t) \exp\left(\frac{\eta}{K} y_i(t)\right). \tag{2.7}$$

Using well-known inequalities, $e^x \leq 1 + x + \frac{1}{2}x^2$ for $x \leq 0$, and $\frac{1}{2} < e - 2$, we replace the e^x term in Eq. (2.7) by $e^x \leq 1 + x + (e-2)x^2$, giving us the following inequality.

$$\begin{aligned} \frac{W_{t+1}}{W_t} &\leq \sum_{i=1}^{N} q_i(t) \left[1 + \frac{\eta}{K} y_i(t) + (e-2)\left(\frac{\eta}{K} y_i(t)\right)^2\right] \\ &\leq \left[1 + \frac{\eta}{K} \sum_{i=1}^{N} q_i(t) y_i(t) + (e-2)\left(\frac{\eta}{K}\right)^2 \sum_{i=1}^{N} q_i(t) {y_i}^2(t)\right]. \end{aligned}$$

Setting $x = \frac{\eta}{K}\sum_{i=1}^{N} q_i(t)y_i(t) + (e-2)\left(\frac{\eta}{K}\right)^2 \sum_{i=1}^{N} q_i(t){y_i}^2(t)$, applying the inequality, $1 + x \leq e^x$, and taking logarithms on both sides, we get

$$\ln \frac{W_{t+1}}{W_t} \leq \frac{\eta}{K}\sum_{i=1}^{N} q_i(t)y_i(t) + \frac{(e-2)\eta^2}{K^2}\sum_{i=1}^{N} q_i(t){y_i}^2(t). \tag{2.8}$$

Now, adding up for $t = 1, 2, \ldots, T$, we get

$$\ln \frac{W_{T+1}}{W_1} \leq \frac{\eta}{K}\sum_{t=1}^{T}\sum_{i=1}^{N} q_i(t)y_i(t) + (e-2)\left(\frac{\eta}{K}\right)^2 \sum_{t=1}^{T}\sum_{i=1}^{N} q_i(t){y_i}^2(t). \tag{2.9}$$

Focusing on the weight update for one expert i we get,

$$w_i(T+1) = w_i(T)e^{\frac{\eta}{K}y_i(T)} = \prod_{t=1}^{T} e^{\frac{\eta}{K}y_i(t)} = \exp\left(\frac{\eta}{K}\sum_{t=1}^{T} y_i(t)\right). \tag{2.10}$$

Since $W_{T+1} \geq w_i(T+1)$ for all experts, i, we get

$$\ln \frac{W_{T+1}}{W_1} \geq \ln \frac{w_i(T+1)}{W_1} = \frac{\eta}{K}\left(\sum_{t=1}^{T} y_i(t)\right) - \ln N. \tag{2.11}$$

Combining Eqs. (2.9) and (2.11), we get the following inequality.

$$\sum_{t=1}^{T}\sum_{i=1}^{N} q_i(t)y_i(t) \geq \sum_{t=1}^{T} y_i(t) - \frac{K\ln N}{\eta} - (e-2)\frac{\eta}{K}\sum_{t=1}^{T}\sum_{i=1}^{N} q_i(t){y_i}^2(t). \tag{2.12}$$

Replacing back $y_i(t)$ by $-\hat{\mathbf{x}}(t).\xi_i(t)$ on the part of left hand side in Eq. (2.12), we get

$$\sum_{i=1}^{N} q_i(t)y_i(t) = -\sum_{i=1}^{N} q_i(t)\hat{\mathbf{x}}(t).\xi_i(t) = -\sum_{i=1}^{N} q_i(t)\sum_{j=1}^{K}\xi_i^j(t)\hat{x}_j(t). \tag{2.13}$$

We know that $\xi_i^j(t) = 1$ if $i_t = j$, and 0 otherwise. Using the value of $\sum_{i=1}^{N} q_i(t) \times \xi_i^j(t)$ from Eq. (2.2) and replacing it in Eq. (2.13), we get:

$$\begin{aligned}
\sum_{i=1}^{N} q_i(t) y_i(t) &= -\sum_{j=1}^{K} \left(\sum_{i=1}^{N} q_i(t) \times \xi_i^j(t)) \right) \hat{x}_j(t) \\
&= -\sum_{j=1}^{K} \left(\frac{p_j(t) - \eta/K}{1-\eta} \right) \hat{x}_j(t) \\
&= -\sum_{j=1}^{K} \left(\frac{p_j(t)\hat{x}_j(t)}{1-\eta} - \frac{\eta \hat{x}_j(t)}{K(1-\eta)} \right) \\
&\leq \sum_{j=1}^{K} \frac{\eta \hat{x}_j(t)}{K(1-\eta)} \\
&= \frac{\eta \sum_{j=1}^{K} \hat{x}_j(t)}{K(1-\eta)}
\end{aligned}$$

Since only action i_t is chosen at time t, we know that $\sum_{j=1}^{K} \hat{x}_{j(t)} = \hat{x}_{i_t}(t)$. Also, we know that $\eta \leq 1$ and $K \geq 1$ resulting in the following inequality,

$$\sum_{i=1}^{N} q_i(t) y_i(t) \leq \sum_{j=1}^{K} \frac{\eta \hat{x}_j(t)}{K(1-\eta)} \leq \frac{\hat{x}_{i_t}(t)}{(1-\eta)}$$

Summing over $t = 1, 2, \ldots, T$, we get

$$\sum_{t=1}^{T} \sum_{i=1}^{N} q_i(t) y_i(t) \leq \sum_{t=1}^{T} \frac{\hat{x}_{i_t}(t)}{(1-\eta)}. \tag{2.14}$$

Similarly,

$$\sum_{i=1}^{N} q_i(t) y_i(t)^2 = q_i(t) \left(-\hat{\mathbf{x}}(t) \cdot \xi_i(t)\right)^2 \leq \hat{x}_{i_t}(t)^2 \frac{p_{i_t}(t)}{(1-\eta)} \leq \frac{\hat{x}_{i_t}(t)}{(1-\eta)} \tag{2.15}$$

Combining inequalities 2.12, 2.14 and 2.15, we get

$$\sum_{t=1}^{T} \frac{\hat{x}_{i_t}(t)}{1-\eta} \geq \sum_{t=1}^{T} y_i(t) - \frac{K \ln N}{\eta} - (e-2)\frac{\eta}{K} \sum_{t=1}^{T} \frac{\hat{x}_{i_t}(t)}{1-\eta} \tag{2.16}$$

Multiplying both sides with $(1-\eta)$ of the above Eq. and taking expectations, we obtain

$$\mathbb{E}\left[\sum_{t=1}^{T} \hat{x}_{i_t}(t)\right] \geq (1-\eta)\,\mathbb{E}\left[\sum_{t=1}^{T} y_i(t)\right] - (1-\eta)\frac{K \ln N}{\eta} - (e-2)\frac{\eta}{K}\,\mathbb{E}\left[\sum_{t=1}^{T} \hat{x}_{i_t}(t)\right]. \tag{2.17}$$

Since $\mathbb{E}\left[\sum_{t=1}^{T} \hat{x}_{i_t}(t)\right] = \sum_{t=1}^{T} x_{i_t}(t) = C_{\text{EXP4-DFDC}} \geq C_{best}$, we get the following inequality.

$$C_{\text{EXP4-DFDC}} \geq (1-\eta)\,\mathbb{E}\left[\sum_{t=1}^{T} y_i(t)\right] - (1-\eta)\frac{K\ln N}{\eta} - (e-2)\frac{\eta}{K}C_{best}. \tag{2.18}$$

Multiplying both sides of the Eq. (2.18) with -1, and using the fact that $\mathbb{E}\left[\sum_{t=1}^{T} y_i(t)\right] = -\,\mathbb{E}\left[\sum_{t=1}^{T} \hat{x}_{i_t}(t)\cdot\xi_i^{i_t}(t)\right] = -\sum_{t=1}^{T} x_{i_t}(t) \leq C_{best}$, we get

$$\begin{aligned} -C_{\text{EXP4-DFDC}} &\leq (\eta-1)C_{best} + \frac{K\ln N}{\eta} + (e-2)\eta C_{best} \\ &\leq (\eta-1)C_{best} + \frac{K\ln N}{\eta} + \eta C_{best}. \end{aligned} \tag{2.19}$$

Finally, adding C_{best} on the both sides of the Eq. (2.19), we get

$$C_{best} - C_{\text{EXP4-DFDC}} \leq (\eta - 1 + \eta + 1)\eta C_{best} + \frac{K\ln N}{\eta} \leq 2\eta C_{best} + \frac{K\ln N}{\eta}.$$

For a fixed time horizon T, since no action can result in a cost greater than 1, $C_{best} \leq T$, thus giving us the following bound on the regret, $R_{\text{EXP4-DFDC}}(t)$, for the EXP4-DFDC algorithm, completing the proof:

$$R_{\text{EXP4-DFDC}}(T) \leq 2\eta T + \frac{K\ln N}{\eta}. \tag{2.20}$$

□

Picking the optimal learning rate for vanishing regret: Theorem 2.4.2 shows that the cumulative regret is a function of the learning rate. We argue that picking the right learning rate can minimize regret and ensure that it vanishes. To minimize the regret, we differentiate the right-hand side of Eq. (2.20) and set it to 0. Thus the equation, $R'_A(T) = 2T - \frac{K\ln N}{\eta^2} = 0$, gives us the optimal value of η as follows:

$$\eta_{OPT} = \min\left(1, \sqrt{\frac{K\ln N}{2T}}\right). \tag{2.21}$$

Finally, plugging this back into Eq. (2.20), gives us the following regret bound:

$$R_A(t) \leq 2\sqrt{2KT \ln N}. \tag{2.22}$$

The significance of this regret bound is that as $T \to \infty$, the regret bound vanishes with the time horizon.

2.4.3 Application to Cache Replacement - Analysis of the LeCaR algorithm

In this section, we show how to use the EXP4-DFDC algorithm and its theoretical analysis from Section 2.4.2 to improve a state-of-the-art cache replacement algorithm. The recent LeCaR algorithm of Vietri et al. [2018] is an outstanding cache replacement algorithm that is based on reinforcement learning and regret minimization. The algorithm accepts a stream of requests for memory pages and decides which page to evict from a cache when a new item is to be stored in the cache following a "cache miss". LeCaR has been shown to be among the best performing cache replacement algorithms in practice [Vietri et al., 2018]. Experiments have shown that it is competitive with the best cache replacement algorithms for large cache sizes, and is significantly better than its nearest competitor for small cache sizes including the state-of-the-art methods like ARC, which was designed over 15 years ago by Megiddo and Modha [2003, 2004]). The LeCaR algorithm is an online reinforcement learning algorithm that relies on only two very fundamental cache replacement policies typically taught in an introductory Operating Systems class, namely the *Least Recently Used* (LRU) policy and the *Least Frequently Used* (LFU) policy. LeCaR assumes that the best strategy at any given time is a probabilistic mix of the two policies and attempts to "learn" the optimal mix using a regret minimization strategy. Thus,

the LECAR algorithm can be thought of as a special case of MAB with two arms, but with delayed feedback and decaying costs.

If an evicted page is requested again, then some "regret" is associated with the eviction decision that caused the miss. However, the regret is greatest if it is requested immediately after an eviction. As the gap between when the page is evicted and when it is requested again increases, the regret decays, until a certain threshold on the delay, beyond which the regret is assumed to drop to zero. Regret is assigned to the policy that caused the eviction of that entry. In other words, LECAR attempts to sense which of the two arms is likely to result in less regret at any given point in the request sequence, and also successfully sensing when the tide may be changing to a different arm. One of the primary shortcomings of LECAR is that its learning rate had to be fixed. Vietri et al. [2018] experimented with different learning rates and picked one that worked best for the data sets they used. Although the data showed that different (fixed) learning rates resulted in the best performance for different data sets, they identified a value that worked best for (most) of their data sets. We show that the LECAR algorithm, when modified appropriately, is a version of the EXP4-DFDC algorithm, allowing us to use the results of Section 2.4.2 and proving that a modified version of LECAR has vanishing regret over time. For the rest of this dissertation, the version of LECAR with minor modifications will be referred to as OLECAR, which stands for LECAR with *optimal* learning rate.

The following notation will be used for the discussion below. Let N denote the number of experts. In LECAR only two experts were exploited – LFU and LRU. Let K denote the number of possible decisions to choose from. In LECAR, this is bounded by the cache size (plus 1 for "no action"), since the decision refers to which item to evict. Since LECAR manages a first-in-first-out history data structure to track the most recent evictions, the delay in feedback, denoted by d, is "approximated" by its position in the history data structure. If the entry is present in history, the delay is bounded by the size of the history, h.

If it is not present, then the feedback is ignored, with the assumption that regret is negligible. The delay value for the current request is used to update the weights of the experts in each round. The user-defined learning rate is denoted by η. The cost of an eviction in LECAR was calculated as $x_{i_t}(t)^d$, where $x_{i_t}(t)$ was called the discount factor and was set to $0.005^{\frac{1}{K}}$. To conform to the new formulation of EXP4-DFDC, we changed the estimated cost function to be $\frac{x_{i_t}(t)}{d}$ in OLECAR. Both LECAR and OLECAR attempt to minimize the cumulative cost given by the expression, $C_A(T) = \sum_{t=1}^{T} x_{i_t}(t)$. This, in turn, attempts to minimize the regret function given by the expression, $R_A(T) = C_{best}(T) - C_A(T)$.

Algorithm 2: The OLECAR algorithm

Input: Request sequence, σ; Cost vectors, $\mathbf{x}(t)$; Learning rate, $\eta \in (0, 1]$; History size, h; Cache size, K; set of experts $= \{LFU, LRU\}$

begin

Set $N = 2, w_1(1) = w_2(1) = 1$, and $W_t = \sum_{i=1}^{N} w_i(t)$;

for $t = 1, 2, \ldots$ **do**

Obtain expert advice $\xi_1(t), \ldots, \xi_N(t)$ for request $\sigma(t)$;

Select action $j \in \{1, \ldots, K\}$ with prob,

$p_j(t) = (1 - \eta) \sum_{i=1}^{N} \frac{w_i(t) \times \xi_i^j(t)}{W_t} + \frac{\eta}{K}$;

for $i = 1, \ldots, N$ **do**

$$\hat{x}_{i_t}(t) = \begin{cases} \frac{x_{i_t}(t)}{d}, & \text{if } i_{t'} = j \text{ and } \sigma(t') \text{ is in history in position } d \\ 0, & \text{otherwise} \end{cases}$$

$$w_i(t+1) = w_i(t) \exp\left(\frac{-\eta \hat{\mathbf{x}}(t) \cdot \xi_i(t)}{K}\right)$$

end

end

end

Next, we formalize the regret bound for the OLECAR algorithm. From the theoretical analysis of EXP4-DFDC, we have an optimal learning rate to choose for OLECAR with theoretical guarantees. Even though the learning rate suggested by Eq. (2.21) decreases with time, the decreasing learning rate works when the time horizon T is known. Furthermore, a decreasing learning rate is feasible when the environment is stationary and

does not change over time once the distribution is learnt. But in dynamic settings such as for the cache page replacement problem, the system is not stationary over time. For this dynamic settings, decreasing learning rate will not work when the system is stationary for a period of time, but learning needs to be restarted as the environment change is detected. As an initial attempt, we set the learning rate at the theoretically optimal value replacing T with 1 and incorporating it in OLECAR. This learning rate depends on the cache size of the system and number of experts. Consequently, we are able to get rid of the fixed choice of learning rate in LECAR, leading us to a new version called OLECAR.

Corollary 2.4.3 *Assume that* OLECAR *(Algorithm 2) is run with a learning rate of* $\eta = \min(1, \sqrt{\frac{K \ln N}{2T}})$*, then the expected regret of the algorithm can be upper bounded by,*

$$R_{\text{OLECAR}}(T) \leq 2\eta T + \frac{K \ln N}{\eta}. \tag{2.23}$$

In this work, we laid the groundwork for a theoretical framework of reinforcement learning in the context of cache replacement. For the cache replacement problem, we developed a novel MAB version with delayed feedback and decaying cost (MAB-DFDC). This version is unlike any of the previous Bandit frameworks. We showed that the EXP4-DFDC algorithm, which incorporates delayed feedback, has vanishing regret properties. Finally, we showed that LECAR and OLECAR machine learning-based cache replacement approaches and a simplified form of EXP4-DFDC, are theoretically guaranteed to have vanishing regret.

CHAPTER 3

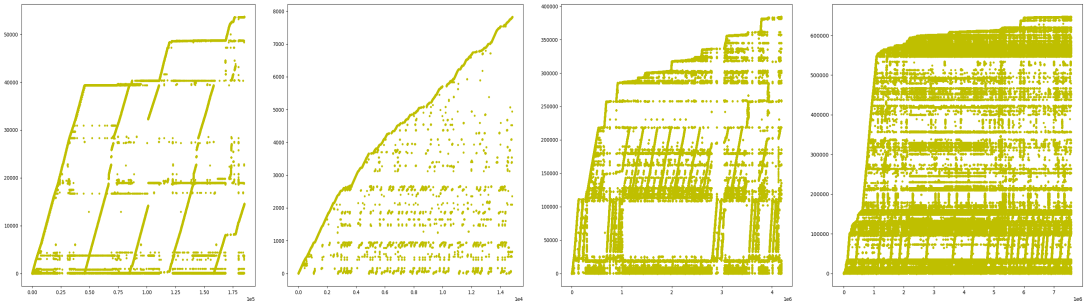
ALECAR: LECAR WITH ADAPTIVE LEARNING RATES

3.1 Motivation

Many domain-specific problems are solved with the aid of machine learning algorithms [Bojarski et al., 2016, Krizhevsky et al., 2012, Mnih et al., 2013, Chen et al., 2017, Beam and Kohane, 2018]. The best cache replacement algorithms in computing systems are *adaptive*, which can be considered as a mild type of machine learning. On the other hand, explicit machine learning methods have only recently been investigated, owing to the fact that machine learning is often computationally intensive, making it difficult to utilize in low-level applications such as cache replacement problem. ML-based frameworks were previously designed (see ACME [Ari et al., 2002], **L**RFU[Lee et al., 2001]), but have had limited success in cache replacement. ACME employs a pool of strategies, resulting in high costs of maintaining expensive information without an appreciable performance benefit, making its viability questionable. **L**RFU showed promising performance exploiting only two fundamental policies (**L**RU,**L**FU), but no further improvements were carried out to tune the configurable parameters, which are termed *hyperparameters*.

The machine learning-based LeCaR [Vietri et al., 2018], an online adaptive cache replacement algorithm, has been recently proposed to efficiently facilitate the cache replacement policy. LeCaR algorithm demonstrated that exploiting just two essential strategies, LRU and LFU, was sufficient to outperform the state-of-the-art ARC algorithm for small cache sizes and competitive for large cache sizes on a limited set of experiments. Machine learning algorithms typically contain many built-in hyperparameters. LeCaR is no exception; It uses a *learning rate* to set the magnitude of the change when the algorithm makes a poor decision and regret is accrued. In LeCaR, the learning rate was chosen to be fixed at 0.45 empirically [Vietri et al., 2018]. However, experiments with a broad range

of workloads showed that different values of the learning rate were optimal for different workloads (Fig. 3.3) [Rodriguez et al., 2021]. LeCaR achieves good performance but lacks the generalization since it also has a static hyperparameter, learning rate (η), that may need to be set beforehand. Fig. 3.3 shows the results of our experiments on four different workloads with varying learning rates. The results showed that the performance varies significantly, and therefore, LeCaR is highly sensitive to the chosen value of a fixed learning rate. Moreover, Fig. (3.3) indicates that with the optimal learning rate, which is unknown, LeCaR can achieve better performance. Furthermore, the request pattern could change quickly within the same workload (Fig. 3.1), requiring different learning rates at different time points.



(a) FIU workload. (b) Nexus workload. (c) MSR workload. (d) CloudVPS workload.

Figure 3.1: Access pattern for different workloads from different sources: Requested block address as a function of time.

LeCaR is sensitive to the input pattern of the workloads since it exploits two basic cache replacement algorithms (LRU and LFU) and inherits their limitations on handling workload-specific patterns. Finally, experiments with LeCaR (Fig. 3.4 and Fig. 3.4) have shown that high learning rates lead to overly quick responses to small changes leading to potentially instabilities. On the other hand, low learning rates can delay much needed reactions to changes in the pattern. Finding a proper balance between these two extremes is necessary for good performance across different input patterns. In the algorithm, OLeCaR, presented in Chapter 2, we fixed the learning rate depending on the cache size and the

number of experts (two experts in LeCaR). It is therefore not surprising that OLeCaR is also not responsive enough to detect the changes in input pattern as it remains fixed for a particular cache size. This chapter focuses on adapting the learning rate in LeCaR and OLeCaR to respond to the changes in the input patterns and presenting an improved, adaptive, machine learning algorithm called ALeCaR.

3.2 Background

3.2.1 Caching algorithms

In this section, we first briefly describe different cache replacement algorithms and their limitations. Different page replacement algorithms exploit different workload features, i.e., recency, frequency, and reuse distance (time difference between two consecutive accesses to the same item) of the requested sequence to evict an item from the cache. The main goal is to increase the number of hits in the cache for the requested items resulting in better performance. The metric used for evaluating the performance of the caching algorithm is typically cache hit rate, a percentage representation of the number of cache hits to the number of total requests.

Least Recently Used (LRU)

The *Least Recently Used* (LRU) algorithm uses a greedy approach in which the algorithm replaces the page used least recently. The concept is based on the fact that the least recently used item is unlikely to be requested again shortly and termed *locality of reference*. LRU is not scan-resistant since it allows a *scan* (one-time sequential requests with the temporal locality) type request stream to pollute the cache, which is a major bottleneck.

Least Frequently Used (LFU)

The The *Least Frequently Used* (LFU) algorithm keeps track of the frequency of a requested item. When the cache is full and a new item is requested, the LFU algorithm will replace the item with the lowest reference frequency. LFU is unable to capture the temporal feature of the workloads. As a result, new items entering into the cache are at risk of being replaced very quickly, as they have a low frequency at first, even though they might be re-accessed soon.

Adaptive Replacement Cache (ARC)

The adaptive caching algorithm, *Adaptive Replacement Cache* [Megiddo and Modha, 2003], utilizes both the recency and frequency of the referenced item. ARC divides the cache into two LRU "subcaches", $T1$ and $T2$. $T1$ contains the items that have only been referenced once, while $T2$ includes the items requested more than once (i.e., items with frequency more than 1). It also maintains a history of recently evicted items into two LRU (FIFO) lists $B1$ and $B2$, containing items evicted from $T1$ and $T2$, respectively. When an item is found in $B1$ (resp. $B2$), it increases (resp. decreases) the size of $T1$, thus, decreasing (resp. increasing) the size of $T2$. ARC is scan-resistant; Unlike LRU, it utilizes the $T1$ to pass through the scan requests without excessively polluting the cache. Unfortunately, ARC is unable to capture the complete frequency distribution of the workloads due to its use of an LRU list for $T2$ and does not generalize for workload where LFU algorithm performs well.

Low Interference Recency Set (LIRS)

LIRS [Jiang and Zhang, 2002] is a well-designed caching algorithm based on the concept of reuse distance. LIRS dynamically ranks the accessed items as either low inter-reference item or high inter-reference item and maintains two stacks S and Q to hold them. LIRS

efficiently manages scan type sequences by routing one-time accesses into its Q filtering list of small size. However, since LIRS uses a fixed-length Q, its ability to adapt is limited. Moreover, like ARC, LIRS does not have access to the entire frequency distribution of accessed objects, limiting its effectiveness for workloads where LFU performs best.

Dynamic LIRS (DLIRS)

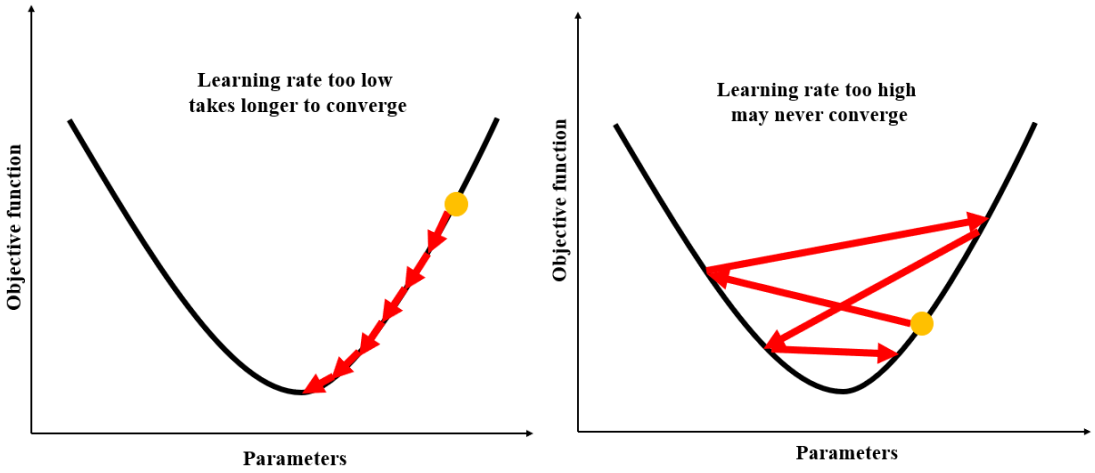
DLIRS [Li, 2018], a recently proposed caching policy, adapts LIRS to adjust the size of cache partitions for high and low reuse distance items. Although this strategy achieves comparable performance to ARC on LRU-favorable workloads for specific cache size configurations while retaining the scan-resistant properties of LIRS, its performance showed inconsistency across the workloads [Rodriguez et al., 2021]. Finally, as with LIRS, it fails to perform well for the LFU-favorable workloads.

Learning Cache Replacement (LeCaR)

As described in Chapter 2, LeCaR [Vietri et al., 2018] is a machine learning-based caching algorithm that employs reinforcement learning and regret minimization techniques to dynamically learn the best eviction strategy from two simple experts, LRU and LFU. An expert is selected at random for each eviction, with probabilities proportional to the experts' weights, w_{LRU} and w_{LFU}. It also maintains separate eviction histories for the two experts, h_{LRU} and h_{LFU}. When there is a cache miss for an item, if it is found in one of the histories, LeCaR penalizes the particular expert and decreases the weight of that expert. For real-world workloads, LeCaR outperformed ARC for small cache sizes [Vietri et al., 2018]. However, it has limited adaptability as it has a static learning rate. It also inherits the limitation of LRU's lack of scan-resistance.

3.2.2 Learning rate adaptation

It is challenging to set the learning rate in traditional machine learning algorithms. The learning rate controls how much the parameter's weight in the model changes in response to the estimated error. As discussed earlier, a value too small can result in a long training process slowly progressing towards the optimal point (local or global), by which time the opportunity may have passed. At the same time, a value too high can result in a sub-optimal change if the opportunity is too transitory, or may lead to instability in the learning process (see Fig. 3.2). ML literature has proposed different random search and



(a) Low learning rate.

(b) High learning rate.

Figure 3.2: Impact of low and high learning rate in learning models.

gradient-based approaches for parameter adaptation in learning algorithms [Robbins and Monro, 1951, Smith, 1955, Khachaturyan et al., 1979, Chan and Fallside, 1987, Battiti, 1989, Kirkpatrick et al., 1983, Breiman et al., 2017, Russell and Norvig, 2016, Ganzfried and Yusuf, 2018, 2019]. One [Chan and Fallside, 1987] of the earlier studies investigated the limitations of the use of fixed coefficients and presented an adaptive algorithm using variable coefficients. In contrast to the fixed settings, this adaptive approach was found to be effective and stable, providing easy near-optimal training and avoiding trial and

error choice of fixed coefficients. Randomized approaches to hyperparameter optimization are more effective according to the analytical and theoretical findings when compared against other approaches, i.e., sequential model based, Gaussian approach, tree structured, grid based models [Bergstra and Bengio, 2012, Bergstra et al., 2011]. Many variants of stochastic gradient descents were exploited to tackle the hyperparameter optimization in online settings [Battiti, 1989, Plagianakos et al., 2001]. The back-propagation technique is now the most commonly used in neural network algorithm. However, the algorithm's slow learning speed and local minimum problem are often cited as its major flaws. A dynamic adaptation of learning rate and momentum through oscillation detection for improving back-propagation algorithm efficiency, are suggested by Yoo et al.. Duchi et al. [2011] presented a new class of subgradient methods that dynamically integrate prior knowledge to conduct more informative gradient-based learning. Popular deep learning algorithms have been proposed in the recent years where dynamic adaptation of learning rate and momentum were achieved successfully [Yoo et al., Duchi et al., 2011, Zeiler, 2012, Kingma and Ba, 2014]. Instead of accumulating all past gradients, *Adadelta* [Zeiler, 2012] is a more stable extension of *Adagrad* [Duchi et al., 2011] that adapts learning rates based on a moving window of gradient updates. In contrast to *Adagrad*, it is not necessary to set an initial learning rate in the *Adadelta* algorithm. Finally, one of the successful and popular deep learning algorithms, *Adam* [Kingma and Ba, 2014], focuses on individual learning rates for each parameter and works well both in environments with sparse gradients and in non-convex neural network optimization.

3.3 Experiments

For the cache management application, the input request sequence remains stationary for short periods (due to the principle of *locality of reference*). Still, it's characteristics change

over time as the process generating the requests change. Since the performance of LeCaR [Vietri et al., 2018] depends on the learning rate, η, we first attempted to empirically answer the following questions:

1. What is the sensitivity of LeCaR's performance to the learning rate?
2. What is the sensitivity of LeCaR's parameters to the workload itself? Is it reasonable to expect that the optimal learning rates we found to work with the current workloads will also work with other workloads or will work when the workloads' characteristics change?

3.3.1 Dataset

We collected the production-level workloads from various sources of block I/O traces [SNIA, Arteaga and Zhao], including Microsoft Research Cambridge (MSR) [Narayanan et al., 2008], Nexus Smartphone [Zhou et al., 2015], FIU [Verma et al., 2010], and CloudVPS [Arteaga and Zhao, 2014]) to run the experiments. The sources and their descriptions can be found in Table 3.1.

3.3.2 Sensitivity of learning rate in LeCaR

In order to capture the impact of learning rate in LeCaR, we ran multiple experiments across different FIU [Verma et al., 2010] and MSR [Narayanan et al., 2008] workloads (see Table (3.1) for details on the data sets). Figure 3.3 shows the performance obtained for different values of learning rate within the range 0 to 1 for four different workloads from FIU and MSR data sets. Fig. (3.3) shows that the optimal learning rate depends on the workloads. When the learning rate is close to zero, the performance is low for three of them (A, C, and D), whereas the performance is highest for one of them (B). Figure

Source	# of Workloads	Details
FIU	181	Developer/end user directories Online course management Document store for research projects Mail server Web server
MSR (Cambridge)	22	User Home directories Hardware monitoring Source control Web staging Test web server
CloudVPS	18	VMs on production environment of cloud provider
CloudCache	6	Online Course Webservers
Nexus	16	Booting smartphone process Watching a movie on the smartphone Listening songs on the smartphone Facebook comments Playing video games Watching videos on the YouTube

Table 3.1: Description of datasets used in our experiments

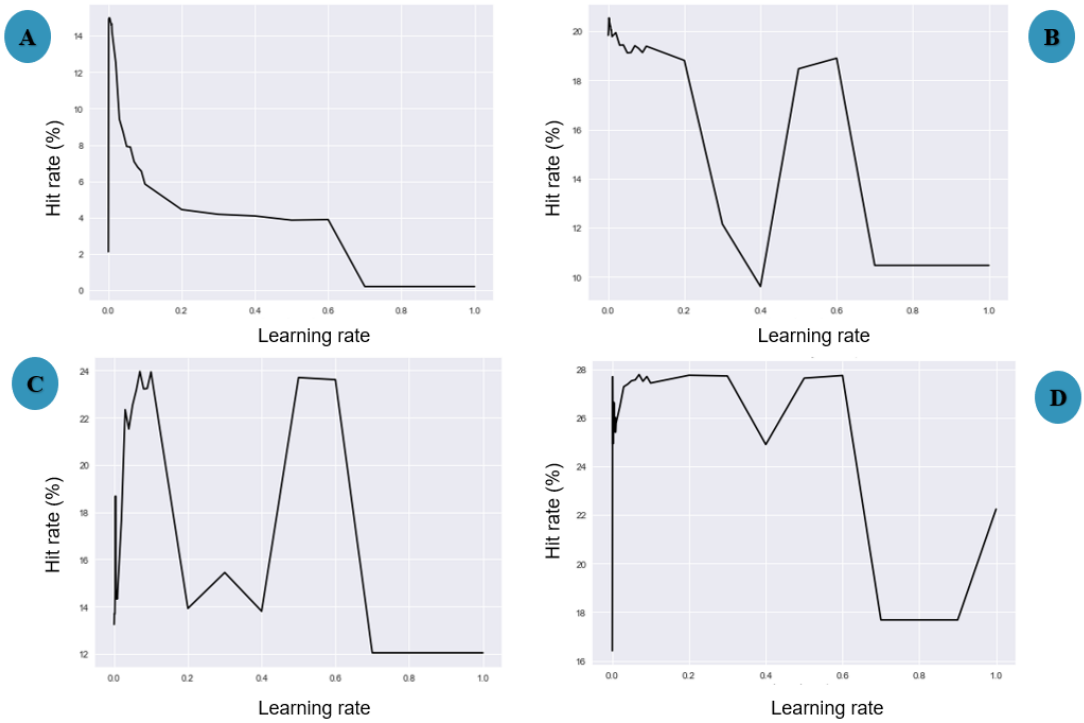
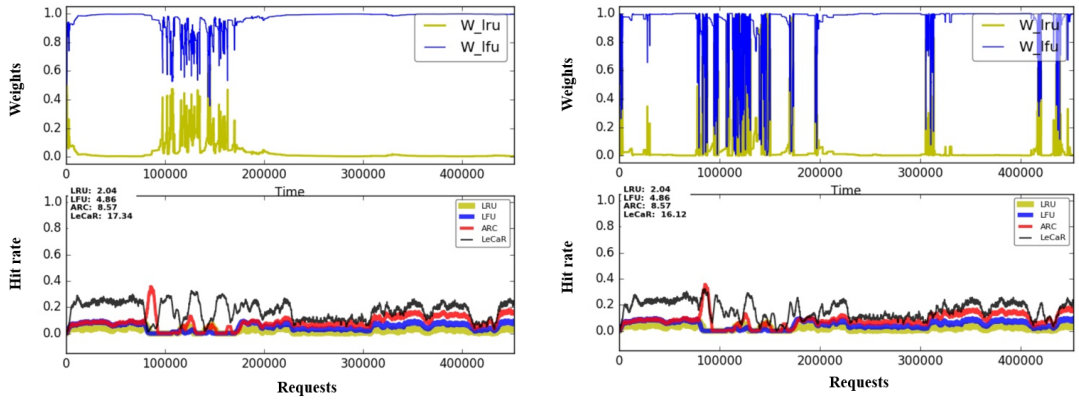


Figure 3.3: Plots of Hit rate against learning rate for the following workloads: (A) FIU Topgun, (B) FIU Ikki, (C) MSR Terminal Server, (D) MSR Firewall Server.

3.3 leads to some interesting observations. In the FIU Topgun (A) workload, if we set the learning rate to zero, LeCaR performs very poorly, but it increases as the learning rate increases to a value close to 0.001. After that, the performance drops. For the other workloads, the performance varies considerably with learning rate. These experiments show clearly that a fixed learning rate is not the best approach for this problem.

Next, we took a closer look at the weights of the experts in LeCaR to the change in learning rates. From Fig. (3.4), the experiments suggest that change in the learning rate can affect the weights drastically in LeCaR. Because the learning rate is used to adjust the weights, for a learning rate close to 0, the weights become stable and change slowly; therefore, lower values can lead to a slow response. Also, it is evident that setting the learning rate too high (near to 1) can cause a quick response to small changes in the data leading to instability.



(a) Low LR (0.1).

(b) High LR (0.9).

Figure 3.4: Impact of learning rates on LeCaR with the weights for LRU and LFU in LeCaR.

In the LeCaR work [Vietri et al., 2018], the learning rate was fixed at 0.45. The experiments suggested that LeCaR is sensitive to the workloads since the optimal value for the learning rate is different for different workloads and that the hit rate vs. learning rate curve can have more than one local optima and/or near-global optima. Therefore, it is possible to make the learning rate dynamic and find the curve's optimum (either local or global) point using stochastic gradient estimation. In summary, any good solution needs to find ways to (a) pick near-optimal learning rates that can be set independent of the workloads, and, (b) for a given workload, must tune the learning rate on-the-fly as the characteristics of the workload changes.

3.3.3 ALeCaR: Adapting the learning rate in LeCaR

To adapt the learning rate in LeCaR, one approach is to design an offline approach that uses historical data to find the peak of the curve from many observations, as was done with LeCaR. This offline approach is easy to implement, but is infeasible because no single optimal value works for every workload, and it depends on the user knowing information

about the workload beforehand. Furthermore, it is a form of *overfitting* and does not guarantee the algorithm's performance on unseen workloads.

A second approach involves optimizing the learning rate at runtime. Online tuning will need careful design to avoid overhead due to computations and memory usage. Thus, while offline optimization requires cross-validation over historical data that may or may not be available, online optimizations come in at least three different flavors. It may be (a) Time-based, (b) State-based, or (c) Gradient-based.

Time-based approach

A high learning rate can result in an unstable fluctuation of weights and prevent the learning algorithm from converging. As a result, the traditional time-based approach starts with a high learning rate initially. It gradually lowers the learning rate over time, keeping it constant for the first T iterations using the following equation: $\eta(t) = \eta(0)/(1 + t/T)$. The *disadvantage* is that it introduces a new hyperparameter, η, that must be set by trial and error, making it unsuitable for online cache optimization.

State-based approach

Another approach involves inferring states and deducing rules from the empirical observations. For this purpose, we can use a *decision tree* approach to deduce states and state transitions. Finally, at each state, it is possible to infer rules for setting and modifying the learning rate. mARC [Santana et al., 2015] uses such state transition to update non-datapath cache policies successfully. Similarly, we can define different states for workload and switch from one to another with different learning parameters at each state. Since it will be rule-based, it would be computationally less expensive and adaptive to the workload at runtime. The main *disadvantage* of this approach is that all the inferences will need to

be made at runtime, which will incur extra overhead in terms of space and time, not to mention the fact that it may not be infer the rules at all times for a given workload.

Gradient-based approach

This leaves the gradient-based approach as the most viable alternative. Popular potential candidates for adaptation in online learning are *Hill climbing*, *Random Restart* [Russell and Norvig, 2016], *Simulated Annealing* [Khachaturyan et al., 1979, Kirkpatrick et al., 1983] and *Stochastic gradient descent* approach [Robbins and Monro, 1951]. Hill climbing is expensive as it explores the vicinity of the point and follows the direction that minimizes the error. It adds an extra overhead of running several instances of the same algorithm, which will be expensive even using a parallel computing model.

In situations where the time to find a better solution is not so critical, simulated annealing could be a good option. However, it is applicable for supervised machine learning models, where immediate feedback is available. Since this is not the case for our problem, all the above-mentioned approaches are not appropriate with the exception of the gradient-based approach, which allows us to update the learning rate [Plagianakos et al., 2001, Duchi et al., 2011] in an adaptive fashion while optimizing for the *hit rate*. The method does not have a provably optimal performance, which is a drawback of this approach. In conclusion, since LeCaR uses a reinforcement learning approach with no prior knowledge of the input, the aforementioned methods are not directly applicable.

Our Hybrid Approach

We considered an approach that combines the time-, state-, and gradient-based approaches. We presented an adaptation approach of learning rate in LeCaR based on time and performance gradient. We also incorporateed the idea of random jump and restart [Smith, 1955]

on top of a traditional stochastic hill-climbing approach to avoid issues like getting stuck on a plateau or in local optima. Several design choices are discussed below.

How often to update: First, we note that we used the hit rate as the performance measure to use for gradient updates. Adaptive algorithms that attempt to improve the performance of computing subsystems require an observation window to collect information so that parameters relevant to the fine-tuning of the system performance can be modified. We choose the size of this window to be the size of the cache. Therefore, we updated the learning rate, η, after N consecutive accesses where N is the cache size. At any time t, the average hit rate over last N accesses at time $t - N$ and t are denoted by HR_t and HR_{t-N}, respectively. Thus, the learning rate (η) is updated at the end of each predefined window.

Which direction to take: The performance (hit rate) is measured for each window. The change in performance during the window is used to decide the direction of update (up or down) for the learning rate. For time t and window N, we track the learning rate used at time $t - 2N$ (η_{t-2N}) and at time $t - N$ (η_{t-N}). If the performance gradient slope with respect to the changes in learning rate is positive at time t, the same direction for the update in learning rate is reinforced for the next window. Else, if the performance gradient with respect to the changes in learning rate becomes negative, the update direction is reversed. Several scenarios can happen as described below.

1. Performance change is positive from $t - N$ to t,
    - Learning Rate changed from $t - 2N$ to $t - N$, follow the last direction to change the learning rate for the next interval.
    - Learning Rate unchanged from $t - 2N$ to $t - N$, make no changes to the learning rate for the next interval.

2. Performance change is negative from $t - N$ to t,

    - Learning Rate changed from $t - 2N$ to $t - N$, reverse the last direction to change the learning rate for the next interval.
    - Learning Rate unchanged from $t-2N$ to $t-N$, try random directions (Possible local minimum).

3. Performance is close to zero for some pre-specified window (Possible plateau),

    - Regardless of the change in learning rate, reset it.

We determined the direction of the gradient, δ_t, using the change in hit rate and learning rate information as follows from Equation (3.1).

$$\begin{aligned} \delta_{HR_t} &= HR_t - HR_{t-N} \\ \delta_{LR_t} &= \eta_{t-N} - \eta_{t-2N} \\ \delta_t &= \frac{\delta_{HR_t}}{\delta_{LR_t}}. \end{aligned} \tag{3.1}$$

Gradient amount

Once the gradient direction is decided, we calculated the update amount for the learning rate at each step. When the learning rate changes from $t - 2N$ to $t - N$, we can take the amount of change, δ_{LR_t}, to quantify the new learning rate in the next interval. However, when the learning rate change is zero and the performance improvement is negative, a random jump is employed to entourage "exploration".

Restart learning to avoid local optimum: The learning rate at each interval is either increased or decreased by a percentage of the previous learning rate (multiplied by the change in learning rate from the previous iteration). However, the rate does not always change. When the learning rate does not change (δ is close to 0) substantially from its

value in recent windows, then there is the possibility that it is stuck in a local optimum. So, whenever the change in learning rate is nearly zero and the performance deteriorates, the learning rate is changed in either the upward or downward direction with random probability to get it out of the local optimum and get the algorithm to start "learning" again. As with many other algorithms, the main challenge is to identify dramatic changes in workload characteristics. Sometimes, even the momentum induced by a random jump may not be large enough to get it out of the local optimum. We wanted to differentiate between the state when the performance is going down due to the characteristics change in workload and the stable state when the learning rate is steady due to high performance. To keep track of the two states, we recorded the performance gradient over the last ten intervals [Einziger et al., 2018]. If the performance keeps going downward, we reset the learning rate to the initial value that was derived from the theoretical analysis of the *Multi-armed Bandit* (MAB) problem (as prescribed in Theorem 2.4.2. The effect of getting stuck in a local optimum can be seen in Fig. 3.5. The green ellipse highlights the region on the hit rate plot where ALeCaR performs worse than the competitors, made worse by a extremely low learning rate during those requests. In this region, ALeCaR has set the weights of LRU to be high and seems to have gotten stuck in a local optimum. Figure 3.6 shows that with restarting the learning, ALeCaR can re-initiate the learning process. We present the learning rate adaptation algorithm incorporating all the steps mentioned above. Algorithms 3 and 4 describe the necessary steps.

Algorithm 3: GradientUpdates(δ_t, δ_{LR_t}, η_{t-N}, η_{t-2N})

if $\delta_t > 0$ **then**
  $\eta_{t-N} = \min(\eta_{t-2N} + |\eta_{t-2N} \times \delta_{LR_t}|, 1)$
else if $\delta_t \leq 0$ **then**
  $\eta_{t-N} = \max(\eta_{t-2N} - |\eta_{t-2N} \times \delta_{LR_t}|, 10^{-3})$

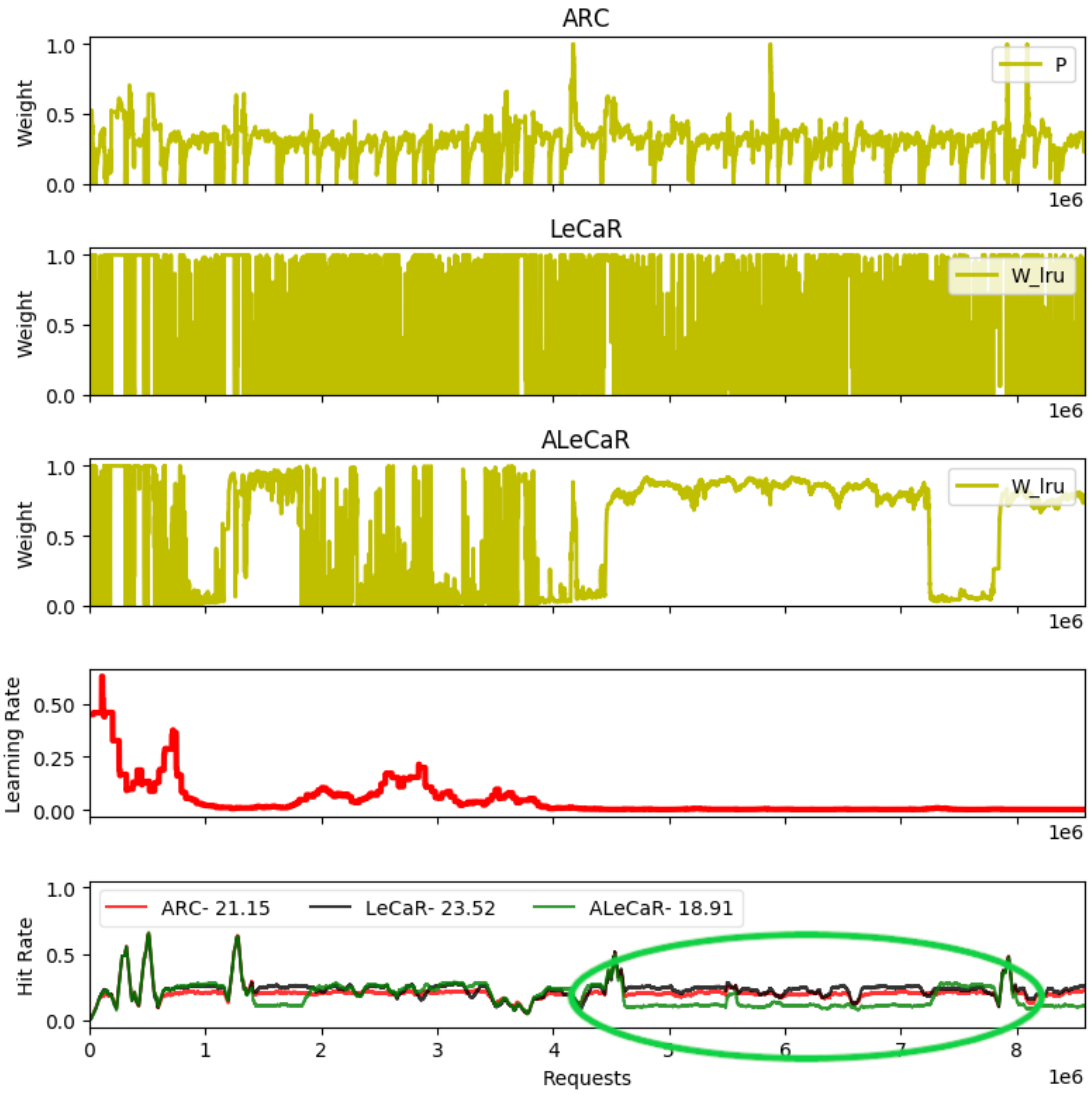


Figure 3.5: Adapting learning rate using a Gradient-based approach without restart; Workload: MSR.

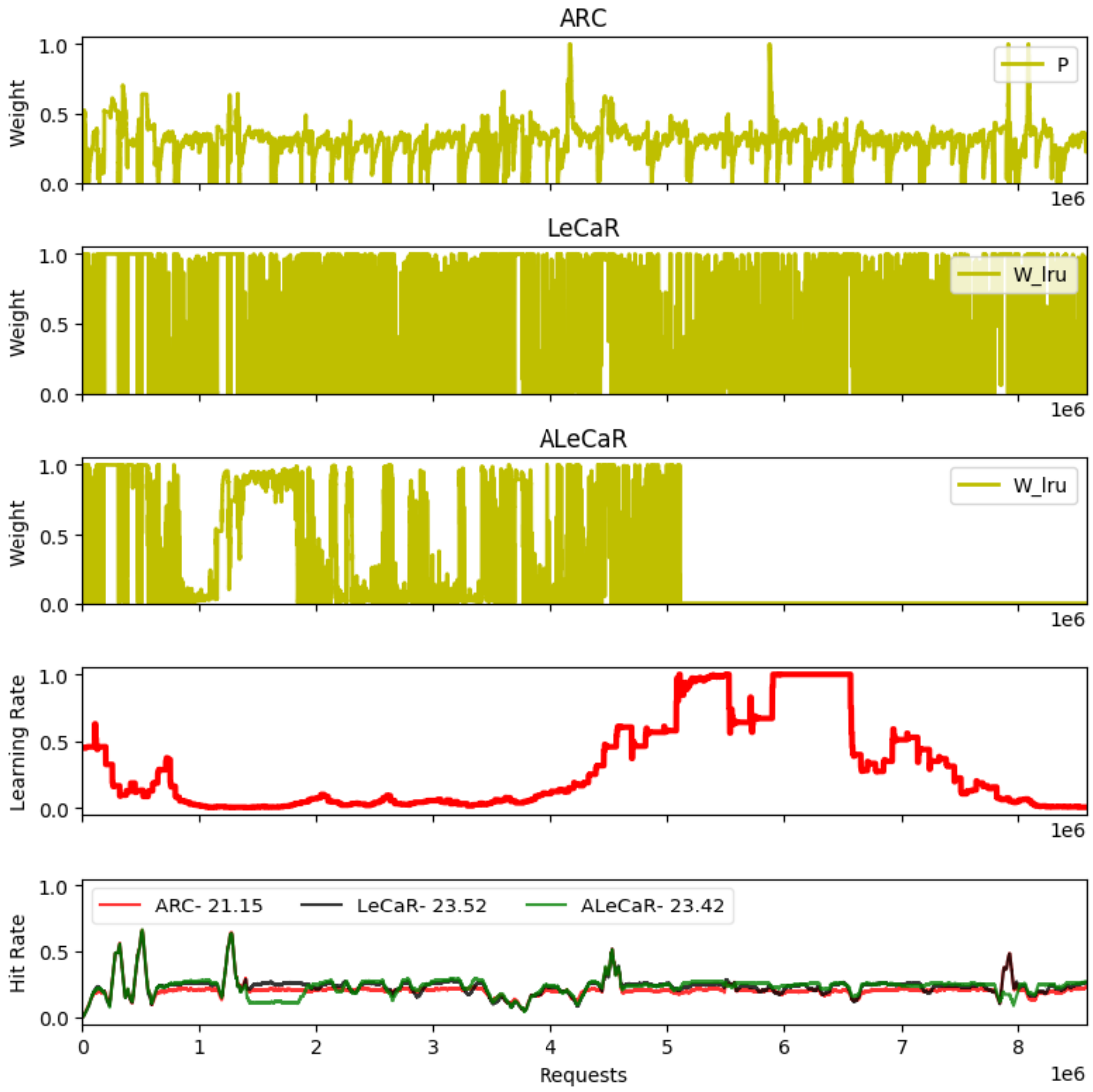


Figure 3.6: Adapting learning rate using a Gradient-based approach with restart; Workload: MSR.

Algorithm 4: UPDATELEARNINGRATE(η_{t-N}, η_{t-2N}, HR_t, HR_{t-N})

$\delta_{HR_t} = HR_t - HR_{t-N}$
$\delta_{LR_t} = \eta_{t-N} - \eta_{t-2N}$
if $\delta_{LR_t} \neq 0$ **then**
    GradientUpdates(δ_t, δ_{LR_t}, η_{t-N}, η_{t-2N})
    $unlearnCount = 0$
else
    if $HR_t == 0$ *or* $\delta_{HR_t} \leq 0$ **then**
        $unlearnCount = unlearnCount + 1$
    if $unlearnCount \geq 10$ **then**
        $unlearnCount = 0$
        δ_{LR_t} = choose randomly between 10^{-3} & 1
        δ_t= Randomly choose to either increase or decrease
        GradientUpdates(δ_t, δ_{LR_t}, η_{t-N}, η_{t-2N})

Algorithm 5: UPDATEWEIGHTS($q, \eta, w_{e_1}, w_{e_2}$)

if $q \in H_{e_1}$ **then**
    $w_{e_1} = w_{e_1} * e^{-\eta}$
else if $q \in H_{e_2}$ **then**
    $w_{e_2} = w_{e_2} * e^{-\eta}$
$w_{e_1} = w_{e_1}/(w_{e_1} + w_{e_2})$ $w_{e_2} = 1 - w_{e_1}$

We adapted the stochastic gradient approach for the learning rate vs. hit rate performance curve and devised an adaptive approach to tune the learning rate in LeCaR. We delineated all the necessary steps: how often to update the learning rate, what direction to take at each step, and finally, with what amount. Algorithm 6 provides the pseudocode for ALeCaR, an adaptive version of LECAR. It employs two experts, LRU and LFU and adjusts the learning rate at runtime. Note that it calls the two algorithms presented in Algorithms 4 and 5.

3.4 Performance analysis

We ran 6,810 simulations on 227 individual workloads within the five sets of workloads, using six different cache sizes. As reported by Vietri et al. [2018], LECAR outperforms

Algorithm 6: ALeCaR(e_1, e_2)

Data: Cache C;
Eviction histories H_{e_1}, H_{e_2};
Weights w_{e_1}, w_{e_2};
Current time t;
Learning rate update window size N;
Learning rate at time t η_t;
Average hit rate at time t HR_t
Input : Requested page q

```
if q ∈ C then
    C.UPDATEDATASTRUCTURES(q)
else
    UPDATEWEIGHTS(q, η, w_{e_1}, w_{e_2})
    if q ∈ H_{e_1} then
        H_{e_1}.DELETE(q)
    if q ∈ H_{e_2} then
        H_{e_2}.DELETE(q)
    if C is full then
        if e_1(C) == e_2(C) then
            C.DELETE(e_1(C))
        else
            action = (e_1, e_2) w/prob (w_{e_1}, w_{e_2})
            if (action == e_1) then
                if H_{e_1} is full then
                    H_{e_1}.DELETE(LRU(H_{e_1}))
                H_{e_1}.ADD(e_1(C))
                C.DELETE(e_1(C))
            if (action == e_2) then
                if H_{e_2} is full then
                    H_{e_2}.DELETE(LRU(H_{e_2}))
                H_{e_2}.ADD(e_2(C))
                C.DELETE(e_2(C))
        C.ADD(q)
if (t%N) = 0 then
    UPDATELEARNINGRATE( η_{t−N}, η_{t−2N}, HR_t, HR_{t−N})
```

the traditional non-parametric approaches (LRU, LFU, ARC) for small cache sizes and is competitive for large cache sizes on a set of experiments done on a limited number of workloads [Vietri et al., 2018]. For ALeCaR, we extended the experiments to include workloads from several sources. We evaluated ALeCaR comparing its performance with

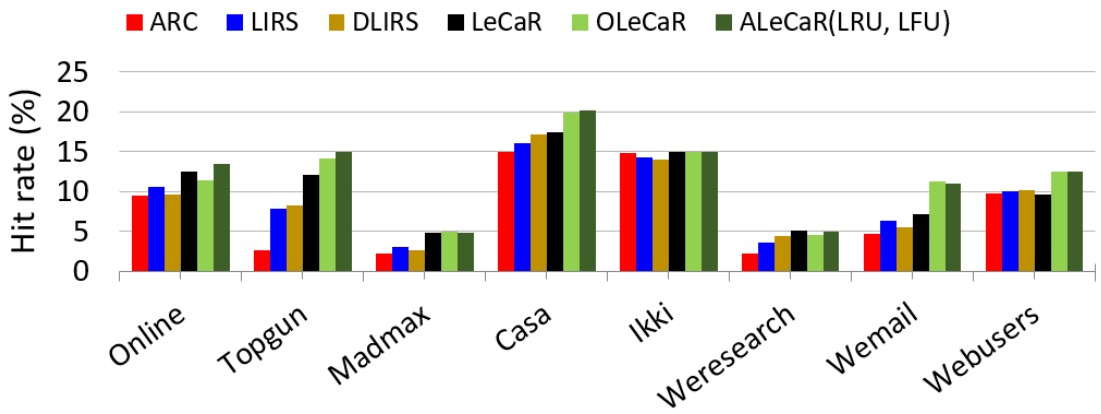
that of ARC, LIRS, DLIRS, LeCaR (with a fixed learning rate of 0.45), and OLeCaR on a small set of workloads. In Figure 3.7, we plot the average hit rate for various cache sizes for the FIU, VPS, and MSR workloads. We note that the adaptive algorithm ALeCaR outperforms ARC for small cache sizes. The performance on FIU workloads shows that ALeCaR outperforms ARC, LIRS, DLIRS, and fixed LeCaR and either better or competitive with respect to OLeCaR. For the MSR and CloudVPS workloads, the average hit rate is worse and is competitive relative to ARC LIRS, DLIRS, and fixed LeCaR for most cases.

Next, we extended our experiment using a large set of empirically collected workloads (see Table 3.1). We excluded the Nexus dataset as they were collected for less than a day, making them unsuitable for our experimets. All the others were one day-long workloads. Also, for further experiments, we excluded the OLeCaR results from our experiment as ALeCaR was either better or competitive in ours experiments. Finally, we plot the hit rate grouping by the cache size. We conducted 4,968 experiments with different cache configurations using those mentioned above four representative traces, i.e., FIU, MSR, VPS, and CloudCache.

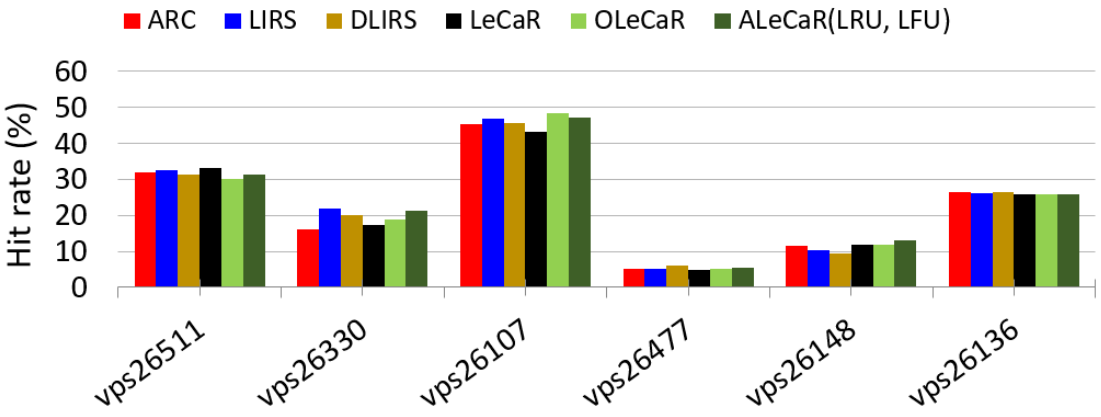
From Fig. (3.8), we observe that the average hit rate for ALeCaR is better than ARC and LIRS for small cache sizes, whereas LIRS outperforms all the algorithms. Also, ALeCaR is competitive with respect to all the algorithms, i.e., arc, LIRS, DLIRS, and LeCaR, for MSR workloads.

Replacing LRU with LIRS in ALeCaR

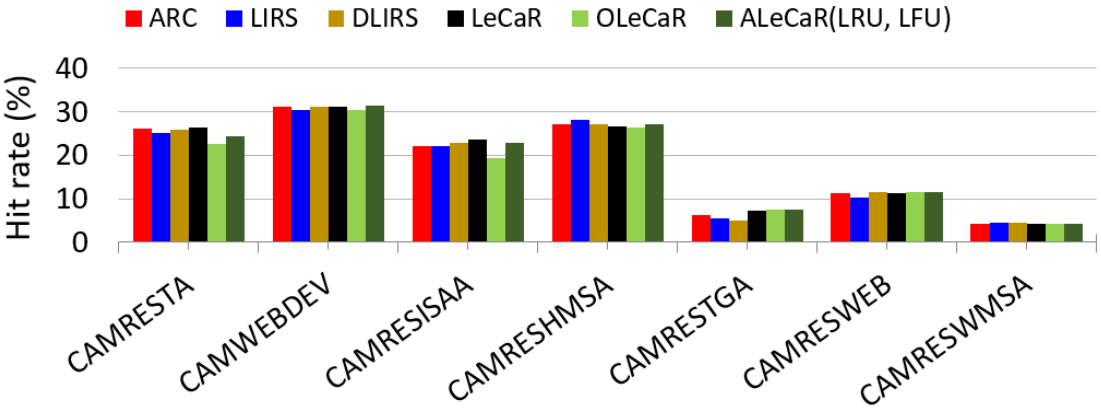
Since the performance of LIRS was better in some cases of our experiments, the pattern indicated that ALeCaR is not doing well for the cases where LIRS performs best. We know that ALeCaR uses LRU and LFU as experts and inherits their limitations. LRU is unable to handle workloads with scan. As a result, ALeCaR also fails to perform well in



(a) FIU workloads.



(b) CloudVPS workloads.



(c) MSR workloads.

Figure 3.7: Average cache hit rate as a function of cache sizes for the following workloads: FIU (top), CloudVPS (middle), and MSR (bottom)

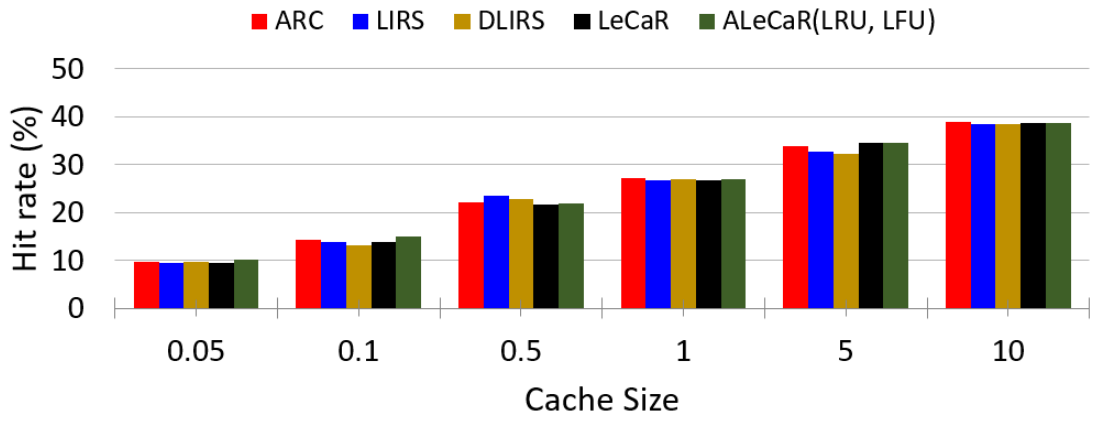


Figure 3.8: Average cache hit rate of 5 algorithms as a function of cache sizes for the FIU workloads for the following algorithms: ARC, LIRS, DLIRS, LeCaR, ALeCaR(LRU, LFU); Cache sizes of 0.1% - 10% of total workload were used for the experiments reported in this Figure.

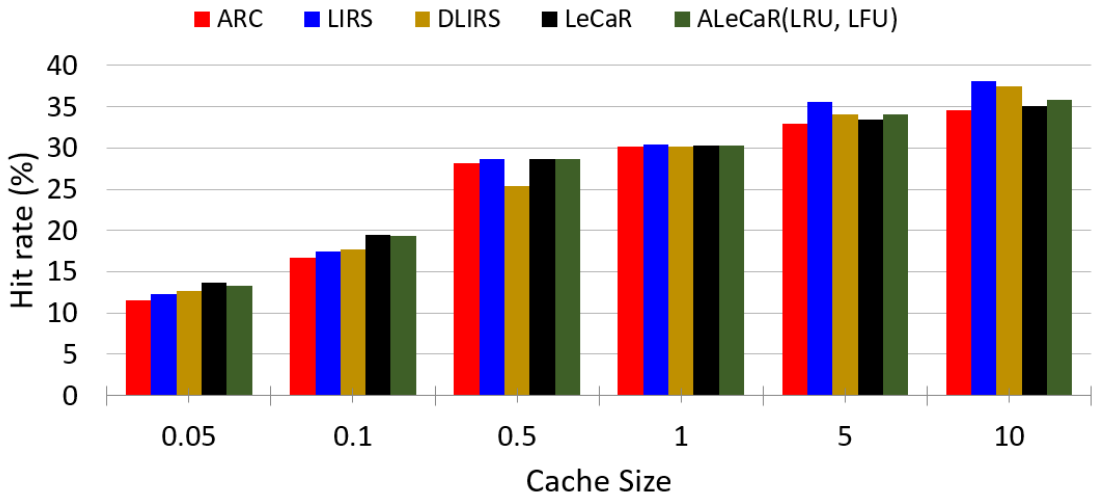


Figure 3.9: Average cache hit rate of 5 algorithms as a function of cache sizes for the MSR workloads for the following algorithms: ARC, LIRS, DLIRS, LeCaR, ALeCaR(LRU, LFU); Cache sizes of 0.1% - 10% of total workload were used for the experiments reported in this Figure.

those cases. Since LIRS overcame the limitation of LRU by introducing scan resistance, we replaced the expert LRU in ALeCaR with LIRS. Next, we plot the performances of all while replacing LRU with LIRS which exhibits the best performances for the different algorithms.

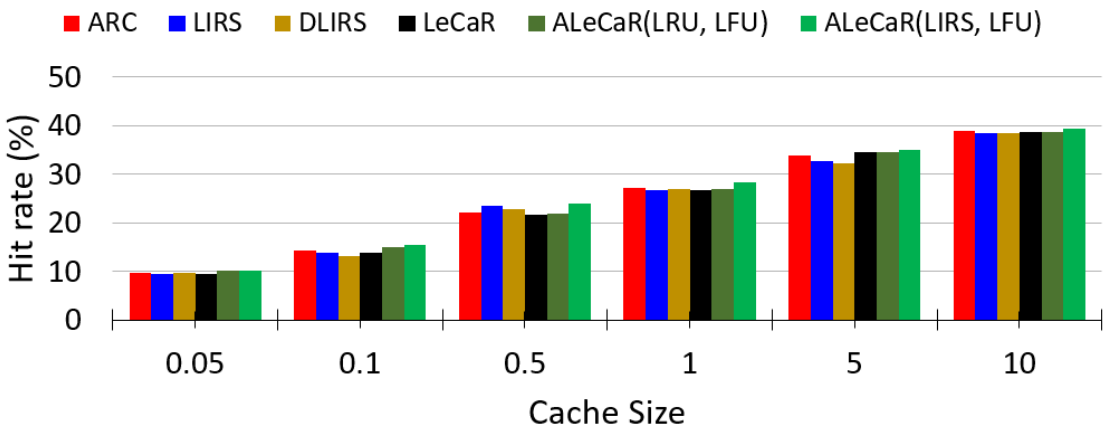


Figure 3.10: Average cache hit rate of 6 algorithms as a function of cache sizes for the FIU workloads for the following algorithms: ARC, LIRS, DLIRS, LeCaR, ALeCaR(LRU, LFU), ALeCaR(LIRS, LFU); Cache sizes of 0.1% - 10% of total workload were used for the experiments reported in this Figure.

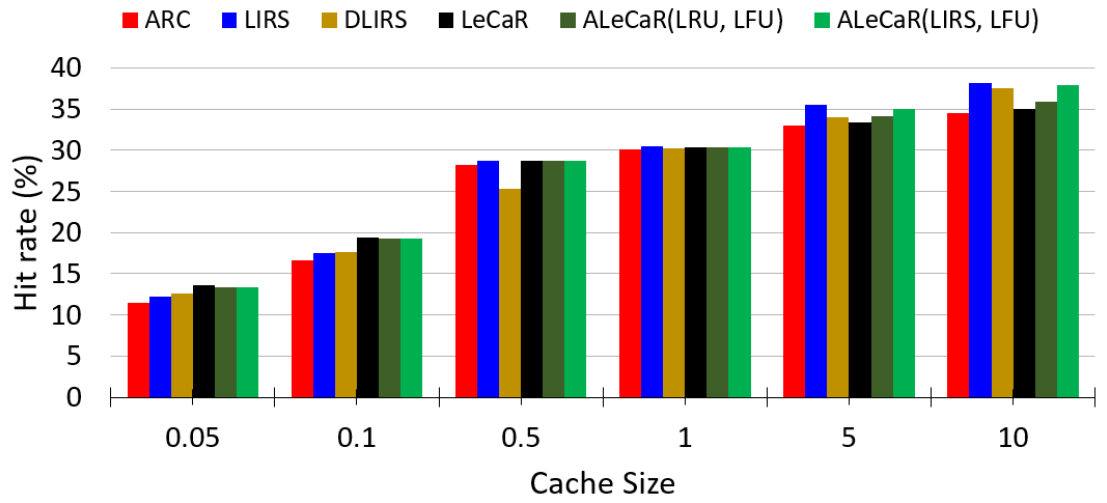


Figure 3.11: Average cache hit rate of 6 algorithms as a function of cache sizes for the MSR workloads for the following algorithms: ARC, LIRS, DLIRS, LeCaR, ALeCaR(LRU, LFU), ALeCaR(LIRS, LFU); Cache sizes of 0.1% - 10% of total workload were used for the experiments reported in this Figure.

Next, we conducted a total of 6,210 experiments with different cache configurations using four representative traces. We also plot the hit rate distribution graph for all the algorithms. Fig. (3.12) displays a violin plot comparing the performance of ALeCaR with the performance of ARC, LIRS, and LeCaR. Positive Y-values indicate that ALeCaR algorithm performs better in comparison. The violin plots show the median as a white dot,

the range from the first to third quartile as a thick bar along the violin's center line, and a thin line showing an additional 1.5 times the interquartile range. It also shows the density shape at each Y-value [Hintze and Nelson, 1998], making these plots very informative. The violin plot is bi-modal for each of the algorithms, which suggests that there exists a broader spectrum of hit rates with two different modes.

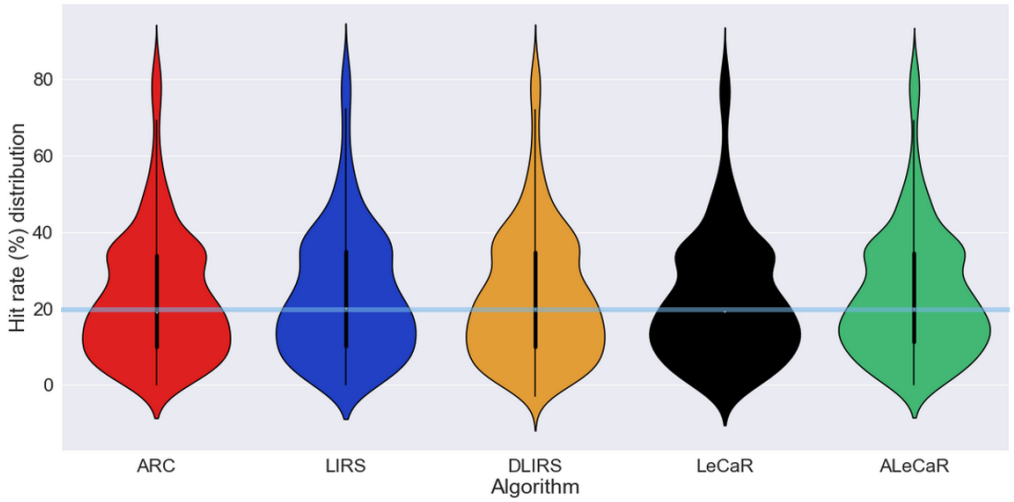


Figure 3.12: Average cache hit rate distribution of 5 algorithms for all the workloads for the following algorithms: ARC, LIRS, DLIRS, LeCaR, ALeCaR(LIRS, LFU); Cache sizes of 0.1% - 10% of total workload were used for the experiments reported in this Figure.

All the experiments we have done so far only looked at the average performance. But, these experiments do not consider the distribution of the hit rates, or the patterns of pairwise differences between two algorithms. Therefore, we performed a broad palette of paired t-tests to evaluate the performance of ALeCaR against the strongest competitors, i.e., ARC, LIRS, DLIRS, and LeCaR grouping them by dataset source and cache sizes. Paired t-test, a statistical method for determining if the mean difference between two sets of experiments on the same datasets is zero, provides the p-value, which can be used to figure out the statistical significance of the difference. The null hypothesis states that the difference between the mean of the paired samples is zero. We used a p-value threshold of 0.05 for this purpose. When the p-value is less than 0.05, we can reject the null hypothesis, which

means the two means are significantly different. In contrast, p-values greater than 0.05 indicate that the difference in the mean values is insignificant. We take this comparison one step further. When assessing the validity of a statistical argument, the *effect size* is also critical as it provides the magnitude of the experimental effect. Therefore, we augmented the significance test with the *effect size* computations. Effect sizes were computed using Cohen's d-measure [Cohen, 2013, Navarro, 2015].

$$\text{Cohen's } d = \frac{(\mu_2 - \mu_1)}{\sqrt{(SD_1^2 + SD_2^2)/2)}},$$

where μ_1 and μ_2 are the means of group 1 and group 2 resp., and SD_1 and SD_2 are the standard deviations of group 1 and group 2 resp. Figure 3.13 presents the results of our t-test analysis for ALeCaR. Each of the boxes in the figure represents both, the effect size (i.e., magnitude of difference between the two averages) and the statistical significance of that difference. Each box compares ALeCaR with the other algorithms for a particular dataset and cache size. The brightness of the color emphasizes the strength of the difference. A brighter color means a larger effect size, and the mean difference is significantly different. Red colors indicate that ALeCaR was significantly worse. The gray color indicates no significant difference. Whereas the brighter green color means ALeCaR was significantly better than the other algorithm, and brighter red colors indicate that ALeCaR was significantly better than the other algorithm. From the figure, we find that ALeCaR performed better than all the algorithms for small cache sizes in the FIU workloads. Also, ALeCaR was significantly better 33% of the times, 17.7% of the times significantly worse, and showing no significant difference 49.3% of the times. The effect sizes range from -0.17 to 1.18. In 40% of the cases, the effect size was negative, and 60% of the times it was positive with 17% of the times the value was greater than 0.2.

In summary, we designed ALeCaR, a new version of LeCaR with improved adaptation of the learning rate. Our evaluation showed that ALeCaR was either better or competitive as the negative effect sizes were much smaller than the positive effect sizes.

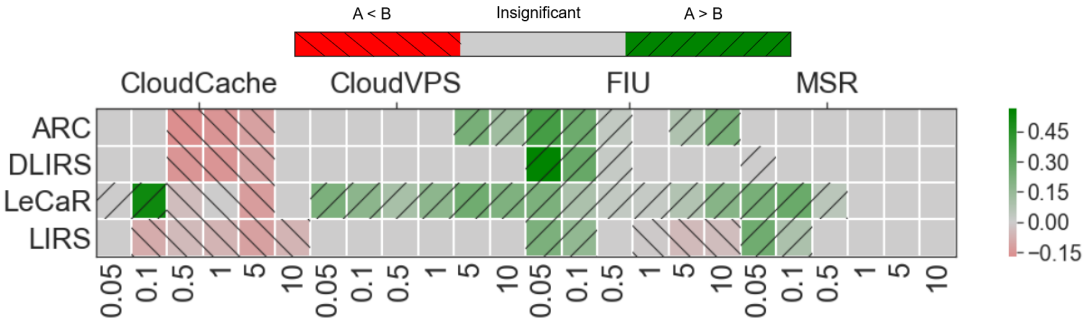


Figure 3.13: Paired t-test analysis to understand the difference in performance between (A) ALeCaR vs. (B) Other.

3.4.1 Interpretability of ALeCaR

One of the key features of ALeCaR is the experts' weights, i.e., LIRS and LFU, which get updated at each round. This feature helps us introduce interpretability to the action taken for future rounds. The algorithm assigns equal weights to both of the experts initially. However, as the learning progresses, the algorithms favor one of the experts depending on the workload characteristics and the current state. As a result, it possible to know what will be the most likely choice of experts in the next round if we look at the weights of the experts at each round. As an example, Fig. 3.5 shows the weights of the experts for different algorithms, i.e., ARC, LeCaR, and ALeCaR, at different rounds. As pointed out in the figure by the highlighted green ellipse, we were able to figure out that both ARC and LeCaR was performing well as the experts were chosen randomly in both cases, but ALeCaR was selecting LRU most of the times and suffered in terms of performance. As a result, we adjusted our adaptation technique to restart the learning at this phase resulting in

better performance for ALeCaR (Fig. 3.6). As a result, we could interpret the weaknesses and strengths of the applied RL-based approach and improve the model further.

3.4.2 Related collaborative work: CACHEUS

Finally, in collaboration with the Systems Research Laboratory (SyLab) at FIU, ALeCaR algorithm was used as a benchmark to develop CACHEUS [Rodriguez et al., 2021], a family of fully adaptive, machine-learned caching algorithms that employ a mixture of experts to handle various types of workloads. State-of-the-art algorithms including ARC, LIRS, and LFU were used along with two new ones – SR-LRU, a scan-resistant version of LRU, and CR-LFU, a churn-resistant version of LFU– were among the experts employed by CACHEUS, which was evaluated using 329 workloads against six distinct cache configurations, resulting in 17,766 simulation experiments. This work showed that CACHEUS, employing the recently suggested lightweight experts, SR-LRU and CR-LFU, is the most consistently performing caching algorithm across a range of workloads and cache sizes, according to paired t-test and effect size analysis. Furthermore, CACHEUS included two more variants where the state-of-the-art algorithms (i.e., LIRS and ARC) were combined with a complementary cache replacement method (e.g., LFU) to manage a broader range of workloads better. The variant with LIRS and LFU is the same as ALeCaR (called C2 in the CACHEUS paper [Rodriguez et al., 2021]). The variant with ARC and LFU is named C1. Finally, the version with two improved experts SR-LRU and CR-LFU is named C3. The performances of the three variants are summarized in Table 3.2.

For storage researchers, development of consistently high-performing caching algorithms remains an intriguing, but elusive, aim. ALeCaR brings us closer to this goal by introducing a machine-learned caching algorithms that is both lightweight and adaptive, which paved the way to the development of the best caching algorithm to date, CACHEUS.

CACHEUS variant	Experts	Effect size range	Better (%)	Not Significant (%)	Worse (%)
C1	ARC, LFU	[-0.62, 0.44]	20	41	39
ALeCaR	LIRS, LFU	[-0.17, 1.18]	26	48	27
C3	SR-LRU, CR-LFU	[-0.31, 2.08]	47	40	13

Table 3.2: Summary of CACHEUS variants statistical analysis.

ALeCaR and CACHEUS [Rodriguez et al., 2021] provided a framework to efficiently and effectively combine cutting-edge caching technique like ARC or LIRS with a complementary expert like LFU to accommodate a larger range of workload primitive types. We believe that machine learning-based frameworks for leveraging caching experts have a lot of promise for increasing the consistency and efficacy of caching systems when handling production workloads.

CHAPTER 4

CAUSAL INFERENCING AND ITS CHALLENGES: THE CASE OF 311 DATA

4.1 Motivation

Innovations in e-governance have enabled citizens and government agencies to join forces in improving the quality of services and achieving greater citizen satisfaction. The 311 contact centers are examples of such innovative systems. The 311 centers are organizations within local governments to field non-emergency service requests. They were enabled by a 1997 Federal Communications Commission policy to reduce the volume of non-emergency calls to 911 centers. The 311 centers have become a hub for local services, to respond to both information and service requests from residents. A citizen can report issues, complaints, or requests for services related to local government. Examples of such service requests include: tree trimming request on a blocked sidewalk, broken stoplight, trash pickup requests, and many more depending on the locality. The requests can be made via mobile apps, social media, online chats, emails and text messages. The service requests are routed to the relevant department through a 311 Customer Relationship Management (311/CRM) system. The data recorded through the 311 CRM system are quite large and granular, which include details about the characteristics or metadata (e.g. location, time, etc.) associated with each service call. As the 311 centers gained popularity over the years, more than 100 cities have implemented them [Thomas, 2013]. With Open 311, a movement toward open data and greater transparency, many cities have made their 311 data public. The 311 data are thus big administrative data sets from local governments. Analysis of the 311 data is useful for policy makers and analysts to gain insights into the nature and demand for services in the local governments.

The purpose of the research reported in this chapter was to apply causal inference methods to the 311 administrative data to gain further insights into the local governments'

response to service requests and to examine if government service provided is uniform across different demographics, socioeconomic classes, and geographical regions. With the ever-growing publicly available administrative data like the 311, it is now feasible to seek answers from the data rather than relying on citizen surveys. The availability of public data has also enabled application of sophisticated models to better understand the causal factors for designing better policies that are effective for a community. In order to make new policy decisions, or figure out the effectiveness of existing policies, it is important to understand the underlying community characteristics that may have a causal influence. Extracting the key characteristics from administrative data provides useful information to policy makers.

The ability to infer causal relationships between variables in a system is an important scientific endeavor. Whereas traditional regression models establish correlations, recent advances in AI have enabled us to develop causal explanatory models. The *causal Bayesian network* (BN) is one such powerful model, which is easy to understand and reasonably interpretable. BNs are probabilistic frameworks that are useful for establishing generic causality and capture more complex, and often, more insightful relationships between variables than a traditional models. The important characteristics of BNs include being able to distinguish between direct and indirect causal dependencies and identifying common effects from the observations [Steyvers et al., 2003]. The theory of *Bayesian networks* (BNs) provides the foundation for us to explore a network of such dependencies. Moreover, these models can simultaneously represent statistically significant knowledge (learned from data) and domain expertise, therefore being the intuitive choice for our causal analysis instead of regression analysis. BNs have been applied successfully in many different fields, such as, gene expressions analysis [Friedman et al., 2000, Zhang et al., 2017, Sazal et al., 2018, 2020a,b], medical services [Margaritis, 2003, Acid et al., 2004], risk assessment and safety systems [Lee and Lee, 2006, Cai et al., 2013, Bayraktarli et al., 2005], epidemiology

[Sachs et al., 2002, Fuster-Parra et al., 2016, Harding, 2011, Ojugo and Otakore, 2020], social sciences [Rigosi et al., 2015, Hudson et al., 2005], econometrics [Tsagris, 2020, Leong, 2016], and more.

In this chapter, we focused on applying the BN to 311 data from one local government: Miami-Dade County in the United States. The county was among the early implementers of a 311 call center in the country. The county's 311 system is operational for all of the unincorporated part of the County (i.e. those areas which have not been incorporated as a city) and most of the incorporated municipalities. For example, the City of Miami, which is located within the county, is also served by the 311 center. Our selection of the county to illustrate the causal model is appropriate since the 311 system is multi-jurisdictional and is among the large well established systems in the country. The 311 center has fielded over 200,000 calls every year in the last five years.

The rest of the chapter is structured as follows. Section 2 provides the necessary background, highlighting the recent related literature. Section 3 outlines the fundamental aspects of Bayesian networks. Section 4 shows the application of BN to the Miami-Dade 311 center data. Section 5 highlights the challenges of applying the BN framework. The Section 6 concludes with the major lessons from the study.

4.2 Background

Local governments (such as cities, counties, school districts, etc.) provide direct public services. Traditionally, residents could request these services only by contacting the appropriate local government agency or department. For example, a resident would need to call the public works department directly to report a pothole. Often, residents would not know which agency should be contacted for obtaining a service. Consequently, the local government services would be available to residents who have the means and the

contacts to local government agencies. Inequities in local government services such as street maintenance, lighting, trash pickup, etc. have been well documented. Wealthier neighborhoods often get better services. Citizen engagement and satisfaction measures were correspondingly substantially lopsided [Welch et al., 2005]. Studies show that the citizens' perceptions and satisfaction toward their government are correlated with accessibility of public services.

With the advent of 311 call centers, any person can call the local government for a service. The person does not need to know which agency fulfills a particular service, and how to contact the agency. The service request is routed at the back end of the 311 CRM system to the appropriate department or jurisdiction. The 311 centers thus arguably democratize the access to local government services, whereby anybody could request services directly without having influential contacts. Residents could make the service requests through multiple methods–over the phone, online (through an app or website), or through social media. Although the 311 expands residents' access to local government agencies through a one-stop method, residents would have to be actually aware of the 311 system's existence and use it. Thus, the empirical question of who calls the 311 center and whether the calls are equitable across different demographic, socioeconomic groups, and geographical areas remains an empirical question. On the flip side, another important question is whether the government agencies provide the services equitably to all sectors of the population. Even though wealthy and low-income areas may make the same service requests, there could be inequities in fulfilling the requests (or their efficiency in doing so) between wealthier and poorer neighborhoods.

Inferring incorrect causal relationships are especially problematic for policy-making in social science since these assumptions often dictate how future investments are made and which groups and areas are prioritized for obtaining public funds and services. The above discussion leads to two crucial research questions. The first question is: Are the service

requests originating equally from the different demographic groups, socioeconomic levels, and geographical areas? This is a question from the demand side, looking at who makes the 311 service requests. The second question is: Are the service requests equitably fulfilled across the groups? This is a question from the perspective of the local government agencies that oversee the service requests. Extant studies have investigated facets of these questions using traditional qualitative and quantitative methods [Clark et al., 2013, 2014, 2020, Elliott and Pais, 2006]. Some of the studies performed hypothesis testing using survey data [Clark et al., 2013, Elliott and Pais, 2006]. One of these papers examines these effects on a wide range of responses, from evacuation timing and emotional support to housing and job conditions and plans to return to pre-storm communities, using survey data from over 1200 Hurricane Katrina survivors. The findings show significant ethnic and socioeconomic disparities. In [Clark et al., 2014], the authors look at service requests made to the City of Boston over the course of a year (2010-2011) and use geospatial analysis and regression analysis to look at potential inequities in service requests based on race, education, and income. The results show that there is no concern that 311 systems (non-emergency call centers) would favor one ethnic group over another. They included race/ethnicity, median income, education, home ownership, and the population as independent variables. Another study [Wang et al., 2017] demonstrated how the 311 service requests recorded and categorized by type in each neighborhood might be used to establish a meaningful classification of locations across the city, based on diverse socioeconomic characteristics, using examples from New York City, Boston, and Chicago. They also show that these traits can be used to forecast future developments in local real estate values. Furthermore, the 311-based classification of urban districts can provide enough data to model. A 15-city review of 311 systems (non-emergency service requests made by city residents) found no systemic disparities in how cities respond that would suggest a bias against minorities and lower-income residents including the independent variables like income, race/ethnicity,

and education [Clark et al., 2020]. However, results were not consistent in all of 15 cities; while some showed no bias, others did exhibit some bias. Based on in-depth interviews with Philadelphia City government officials and managers responsible for creating and operating the 311 center, Nam [Nam and Pardo, 2013] and Hartmann et al. [Hartmann et al., 2017] argue that the program has resulted in a more efficient, effective, transparent, and collaborative city government.

The 311 data has provided new opportunities to gain insights from the quantitative observations [Li et al., 2020]. Open 311 is a standardized protocol which allows for commensurate measurement of 311 service requests across different cities. Predictive models have been applied to extract useful insights from 311 data [Zha and Veloso, 2014, Kontokosta et al., 2017, Xu et al., 2017, Gao, 2018, Xu and Tang, 2020] of different cities, i.e., New York City, Chicago, Philadelphia, etc. These models present analytical frameworks to study overall or particular requests to help better resource allocation and reduce the response time. These studies have shed light on the key features that affect the different types of requests. Different approaches have also been applied to improve service quality in terms of responsiveness at the time of disasters [Madkour, 2020, Zobel et al., 2017]. Elliott et al.'s study of responses after Hurricane Katrina revealed strong biases in providing services by government[Elliott and Pais, 2006]. Xu et al.'s study of 311 service requests after Hurricane Michael in the City of Tallahassee, Florida, also showed similar biases. To date, however, we are not aware of any study that has applied causal learning algorithms to administrative data like 311. The application is important to understand the causal factors that can have substantive policy impacts on how resources should be allocated based on residents' service requests.

It is in the above context that we took advantage of recent advances in machine learning to apply causal learning algorithms to the 311 administrative data set. Since this was among the first such studies to apply the causal methods, this chapter is exploratory in nature. Our

aim was to examine the extent to which the causal methods are applicable to administrative data like 311 for inferring causal relationships in a substantive way. In this process, we also highlighted the challenges that arose in the application of causal methods to the administrative data. As we explain in more detail below, we chose the Bayesian networks method to understand the causal relations.

4.3 Causal Inferencing with Bayesian networks

Bayesian networks (BNs), a class of *Probabilistic Graphical Models* (PGMs) [Koller and Friedman, 2009, Pearl et al., 1995], can be represented as a *Directed Acyclic Graph* (DAG) along with a conditional probability table associated with each node of the graph. The DAG is denoted by $G = (V, A)$, where V is a set of nodes (or vertices) and A is a set of directed edges connecting the nodes in V. Each node in V represents a random variable from a set $\mathbf{X} = \{X_i, i = 1, \ldots, n\}$. A directed edge in A between two nodes reflects a dependency between them. If (v_i, v_j) is a directed edge in G, then v_i is said to be a *parent* of v_j. The conditional probability table associated with v_i. describes the *marginal* distribution of X_i given the joint distribution of the random variables represented by its parent nodes. Each random variable, X_i follows a probability distribution $P(X_i)$, which may be discrete or continuous. The Bayesian network describes the relationships between these distributions. For every node $v_i \in V$, the local probability distribution of its random variable, $P(X_i)$, satisfies the *Markov property*, which states that when conditioned on its parents, X_i is independent of all other variables [Korb and Nicholson, 2010]. An important consequence of the Markov property is that the joint distribution can be written as a product of local conditional probabilities as shown below:

$$P(X) = \prod_{i=1}^{n} P(X_i|Parents(X_i)) \tag{4.1}$$

The above joint probability distribution can be simplified if the BN can be made sparser by eliminating edges. In particular, if two variables are independent or conditionally independent (conditioned on other variables) of each other, then the corresponding edge can be eliminated. Note that if all the variables are independent of each other, then the joint probability distribution is simply the product of the individual distributions, reflecting an empty BN. Also, note that BNs are assumed to be acyclic, i.e., the causal relationships do not form cycles.

The task of fitting a BN is called "model learning". It involves two steps [Scutari, 2014, Scutari and Denis, 2014] as follows:

Structure learning : learning the structure of the network from the data;

Parameter learning : estimation of the local probability distribution implied by the structure learned.

Given a dataset D, if the parameters of the global distribution is denoted by θ, learning the model, denoted by M, can be defined as follows for the graph G:

$$\underbrace{P(M|D) = P(G, \theta|D)}_{\text{model learning}} = \underbrace{P(G|D)}_{\text{structure learning}} \cdot \underbrace{P(\theta|G, D)}_{\text{parameter learning}} \tag{4.2}$$

Here our focus is mainly on *structure learning*. The objective is to find a network (BN) that will encode all the conditional dependencies from the data. If the edges represent relationships that are believed to be causal, we have a causal BN. Structure learning approaches can be grouped into three broad categories: constraint-based, score-based and hybrid. Constraint-based algorithms utilize the probabilistic relations defined by the Markov property of BN, based on the *Inductive Causation* (IC) [Verma and Pearl, 1992] algorithm, which provides a theoretical framework to learn the BN using *conditional independence* (CI) test. The algorithm might not resolve all the directional dependencies from the data; hence, may provide a partially directed acyclic graph (PDAG) [Kalisch et al.,

2012]. A constraint-based IC approach was proposed by Spirtes et al. to learn the BN, [Spirtes et al., 1993]. This approach is better suited for our purposes than the score-based methods as it is known to generate networks with fewer false positives, resulting in a conservative structure in terms of number of edges. *PC-Stable* is an improved order-independent version of IC algorithm [Colombo and Maathuis, 2014, Colombo et al., 2012].

Next we briefly describe the key features of the PC-Stable algorithm. The first step in learning the structure using PC-Stable is to find the pairs of connected nodes in the network. This step starts from a complete undirected graph and eliminates edges using conditional independence tests. This results in a skeleton structure. A 3-node DAG is the smallest structure that can be causally inferred. The basic idea is to use the *Conditional Independence* (CI) test to distinguish between the three possible 3-node DAG structures [Verma and Pearl, 1992]. Also, when the acyclic assumption is relaxed, the structure would fail to generate a cause-effect relationship. The next step is to identify useful substructures, one of which is an important structure called a v-*structure*. Depending on the conditional dependencies between three random variables X_i, X_j, X_k, the useful substructures can be categorized into three different types, as shown in Figure 4.1.

- *Causal chain*: This describes variables that affect each other sequentially. In a BN G, for any three nodes v_i, v_j, v_k, representing variables X_i, X_j, X_k, if there exist directed edges (v_i, v_j) and (v_j, v_k), without the edge (v_i, v_k), then the path: $v_i \rightarrow v_j \rightarrow v_k$, is called a *causal chain*. This represents the case when once X_j is known, X_i and X_k become independent of each other.
- *Common cause*: This represents the case where two variables are impacted by a third variable. The causal structure for this case is as follows: $v_i \leftarrow v_j \rightarrow v_k$. Here, as in the case of the causal chain, once X_j is known X_i and X_k become independent of each other.

- *Common effect*: If the edges are oriented as follows: $v_i \rightarrow v_j \leftarrow v_k$, then it is called a v-structure. The variable X_j is considered to be a "common effect". Here, X_i and X_k are independent of each other, but become dependent when conditioned on X_j. This property makes the v-structure distinguishable from the other two (described above) with the help of a simple CI test.

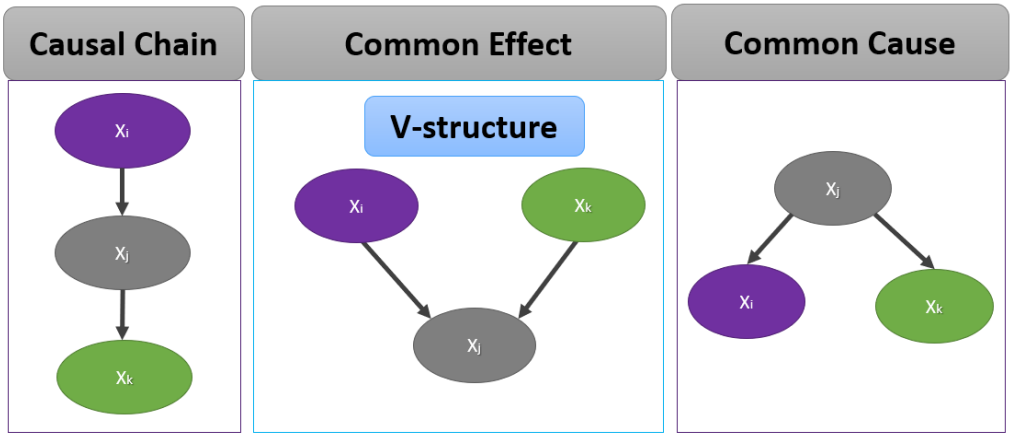


Figure 4.1: Three important causal substructures between variables using the "Conditional Independence" test involving three nodes. A *Causal Chain* suggests only an indirect causation of X_i on X_k. The substructure in the middle is an example of a common effect where one outcome is influenced by two different factors. The substructure on the right is called a common cause structure where there exists two different effects from a single cause.

As mentioned above, causal chains and common causes are not distinguishable with CI tests. As a result, we start by identifying the v-structures and then orient the remaining edges of skeleton to make the network acyclic and consistent. It is possible that the available observational data may not have enough examples to conclude the direction of some edges with statistical certainty. Some algorithms insist on directing all edges even when the direction remains uncertain, while others leave them as undirected edges to reflect the uncertainty.

Finally, the next step of Bayesian learning is parameter learning, where it does regression over the variables learned from the structure and can be used for prediction. Thus, the

approach used in this chapter does use regression, but only on specific subsets suggested by the structure learning step. Note that standard regression techniques do not have the ability to determine the subset of variables on which to apply regression. Our primary focus is to infer the causal structure from the observational data, which would provide us with the subset of variables that affect the dependent variables.

4.4 Causal Inference

4.4.1 Dataset

311 dataset

For the analytics presented here, we downloaded the data from Miami-Dade County's open data portal [Miami-Dade County open data]. The data have been made publicly available to promote quick access and transparency, thus enabling different case studies with the aim of understanding the community better. The dataset includes the service requests from 2013 to present (Fig. 4.2) made by the community member to the 311 center.

The requests were made in one of many ways i.e., through a phone call, app, or website. The breakdown of the volumes of the different types of service requests, mainly dominated by trash pickup requests, is shown in Figure 4.3. Each request record contains key information regarding the type of request, timestamp, and exact GPS location from where the requests were made. This data was combined with the TIGER/Line files and shapefiles from the U.S. Census Bureau for the County, thus associating each request with the the *geographic entity codes* (GEOIDs) at two different geographical granularity levels, i.e., block group and Census tract. The dataset has the Zip code level. We also used the timestamp to calculate the service completion time (number of days taken to complete servicing a request since its initiation). We excluded the observation for which either the

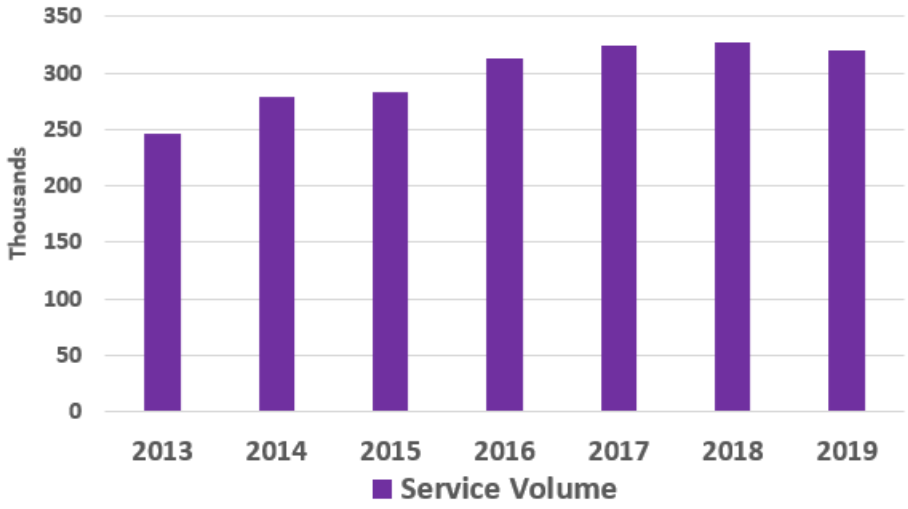


Figure 4.2: Total number of service requests made to the 311 call centers by the local residents aggregated by year ranging from 2013 to 2019 in Miami-Dade County.

request was never closed or the completion time was negative (suggesting a recording error on the timestamp). The services requested were divided into four broad types: requests, complaints, issues, and others. This was based on a keyword search on the issue type. We focused our analysis on the records labeled as "Requests". Records labeled as "complaints" or "issues" typically have longer completion times. Then, we aggregated the total number of requests and the average completion time for each geographic unit. There were 520 and $1,595$ entries in the dataset for the most recent year. Requests aggregated at the Zip Code level were excluded from further analysis since the number of different zip codes in Miami-Dade was too small for useful analyses (79 entries for the latest year).

Census dataset

The 311 dataset does not contain the identity of the person who requested the services. As a result, we combined the geocoded data with the available demographic information for the geographical unit. The demographic information is collected from the 5-year

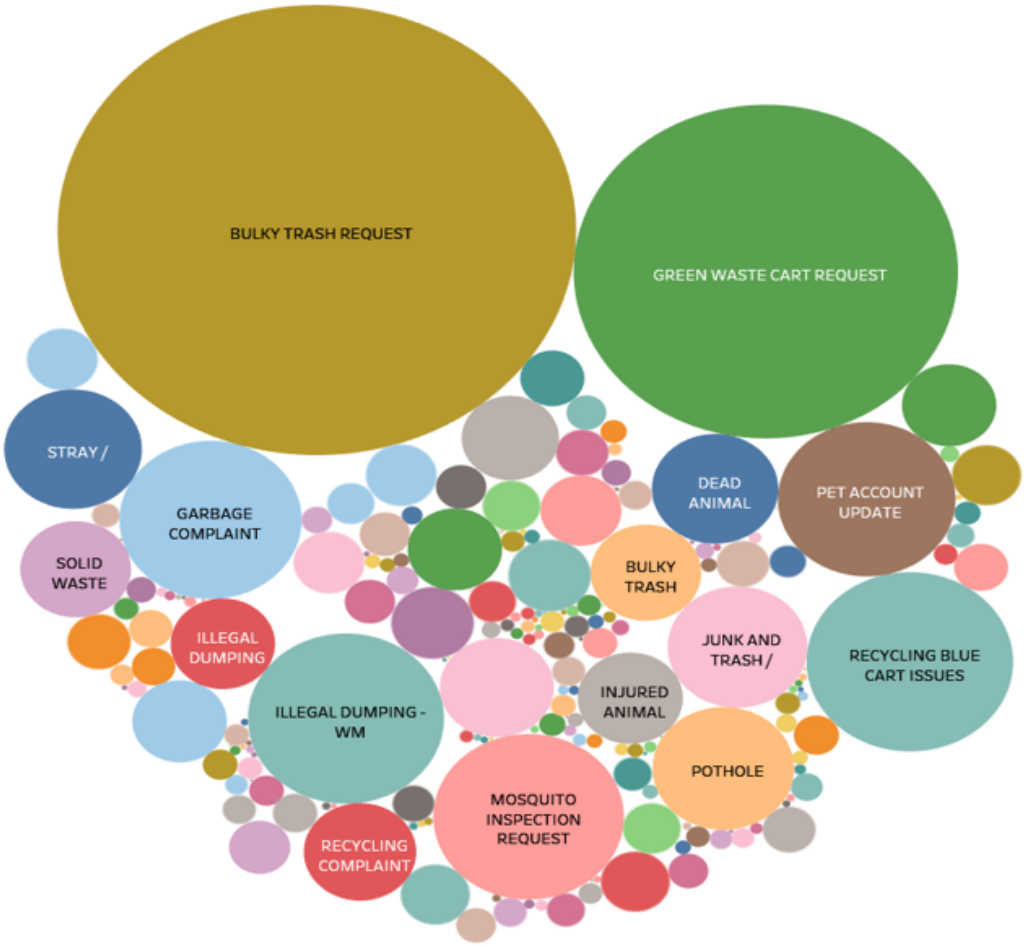


Figure 4.3: Bubble clouds showing all types of service requests for the year 2019 in Miami-Dade County. The count determines the sizes of the bubbles of each service request, and the colors help to distinguish one type of request from the other. Bulky trash pickup and green waste cart request are the top two requests made to the Miami-Dade County by its residents.

Table 4.1: Description of different types of services, i.e., requests, complaints, issues, and others in Miami-Dade County 311 datasets ranging from 2013 to 2019. The completion time (average) is aggregated across different geographical units, i.e., Block group, Census Tract, and Zip code.

	Total No. of records	**Avg. Completion Time**		
		Block group	**Census tract**	**Zip code**
Requests	860,254 (41.15%)	6.82	7.10	8.60
Complaints	133,035 (6.36%)	8.57	8.99	9.43
Issues	110,023 (5.26%)	14.15	14.78	15.32
Others	986,736 (47.2%)	24.02	25.31	27.60
Total	2,090,177	13.29	18.40	18.95

estimates of the American Community Survey (ACS). We conducted analyses of the data at both the Census tract and the block group level to understand the relationships between the measured variables. For this purpose, we included the percentage of the demographic and socioeconomic condition variables from ACS for the year 2013-2019. The average completion time and total request volume (aggregated at the appropriate geographic unit) were used as the target variables. The independent variables considered for this purpose were housing conditions (owner-occupied vs rentals, single unit vs multiple units), race (black vs white), ethnicity (Hispanic vs non-Hispanic), gender distribution (male vs female), and economic condition (unemployed vs employed and below poverty vs above poverty). The details and statistics of the nine independent and dependent variables are described in Table 5.2 and 4.2.

We also provide a matrix of scatter plots of all pairs of variables selected for study in Fig. 4.4, illustrating the pairwise correlations between them. Histograms of all the variables of interest are plotted along the diagonal of the grid (Fig. 4.4). Some of the variables, i.e., 'Owner-occupied units', 'Female population' and 'Unemployed population', exhibit a normal distribution. The derived variables from 311 data and some demographic and socioeconomic variables (i.e., 'Black population,' 'Poor population') show right-skewed

Table 4.2: Census (from ACS data) and derived (from 311 data) statistics of variables for Miami-Dade County. The target variables, volume (total) and completion time (average), were aggregated across different geographical units, i.e., Block group, Census Tract, and Zip code and are separated from the dependent variables by a double horizontal line.

Variable name	**Block group**		**Census tract**		**Zip code**	
	Mean	**SD**	**Mean**	**SD**	**Mean**	**SD**
Volume	114	111	293	309	1627	1853
Completion Time	6.82	5.34	6.55	5.97	8.60	7.69
Owner-occupied units	0.57	0.26	0.59	0.23	0.52	0.20
One unit	0.59	0.35	0.49	0.32	0.47	0.29
Female population	0.51	0.06	0.51	0.04	0.51	0.04
Black population	0.20	0.28	0.19	0.26	0.17	0.21
Hispanic population	0.62	0.28	0.64	0.26	0.56	0.25
Unemployed population	0.42	0.11	0.42	0.09	0.44	0.07
Poor population [Def. by ACS]	0.15	0.12	0.16	0.10	0.14	0.08

long-tail distribution. On the other hand, 'Single housing units' and 'Hispanic Population' in M-DC show left-skewed long-tail distribution. The figure also includes the correlation plots between a pair of variables. Several obersevations can be made from the correlation plots, i.e., 'Owner-occupied units' and 'Single housing units' are positively correlated, 'Owner-occupied units' and 'Poor population shows a negative correlation, 'Unemployed' and 'Poor population' have a positive correlation.

Note that for each type, only one variable was retained. For example, female population was retained for gender, since the male population would be the remainder. Inclusion of both variables introduced extraneous correlations into the network, making inferencing more error-prone and noisy. As another example, the total population is mostly composed of black and white populations.

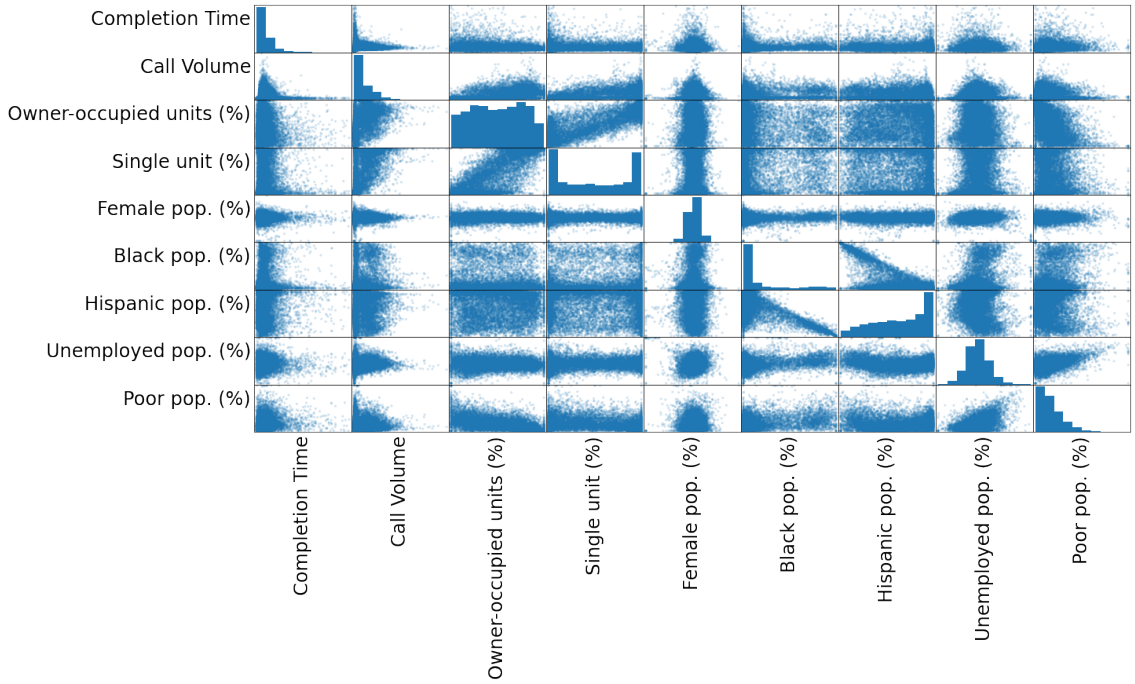


Figure 4.4: Matrix of scatter plots of Census (from ACS data) and derived (from 311 data) variables for Miami-Dade County. The target variables, volume (total) and completion time (average), were aggregated by the block group.

4.4.2 Signed Bayesian network

To generate the *Signed Causal Bayesian network* (sBN) [Sazal et al., 2020a], we followed a two-step approach. First, we applied the PC-stable algorithm, a constraint-based BN learning approach from the `bnlearn` R package [Scutari, 2009]. The algorithm generates an inferred causal network. All the variables are represented as nodes in the network, and each edge is believed to represent a causal dependency. As suggested by Sazal et al. [2020a], we augmented the edges of the network with the help of a *co-occurrence network* (CoNs) [Fernandez et al., 2015]. In CoNs, the edges are colored using a correlation coefficient between the variables (e.g., the Pearson correlation coefficient). A green (red) colored edge means that the correlation between the variables represented by the endpoints is positive (negative, resp.). Finally, the thickness of the edges are determined by the bootstrap strength score [Friedman et al., 2013]. This counts the fraction of times the

edge appears in the network out of a large number of runs. (We used 100 runs for our experiments.) As suggested by Sazal et al. [2020a], we refer to this augmented network as a *Signed Bayesian network* (sBN).

Block group:

The first set of analyses was performed on block level data. The variable representing the volume of requests was categorized as low, medium, or high. The levels were determined using a histogram of values, which suggested a trimodal distribution, as shown in Figure 4.6. Finally, we use the one-hot encoding for the three different categories. No transformation was performed on the other target variable (completion time) since the variable exhibited a unimodal distribution after the outliers were excluded.

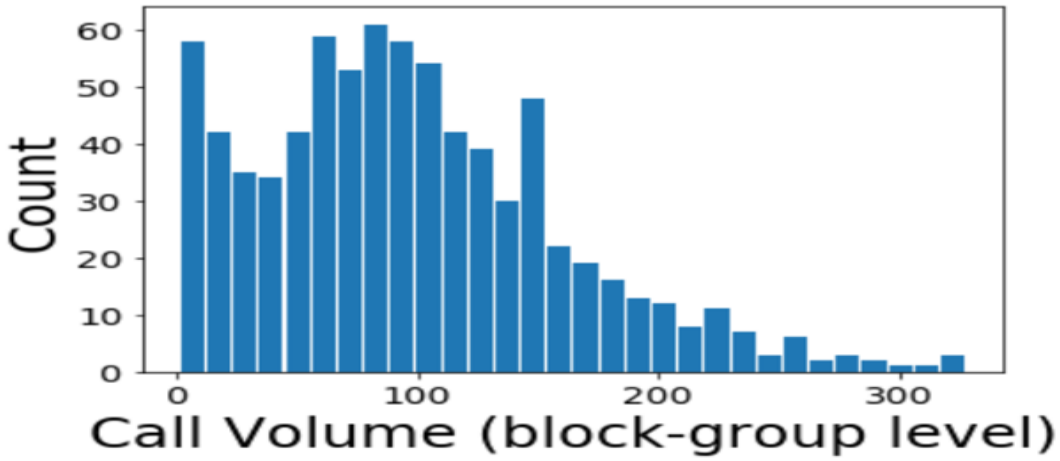


Figure 4.5: Histogram of service request volumes aggregated by the block group level in Miami-Dade County for 2013. The number of bins is 30, and the records having a volume of more than 300 have been discarded.

There were a total of 15 directed edges in the network (Fig. 4.7). The edges were augmented in terms of their color and thickness as described earlier. The target variables had no incoming edges suggesting that none of the variables influenced the volume or completion time. The edge between Completion time and Low volume is undirected. There

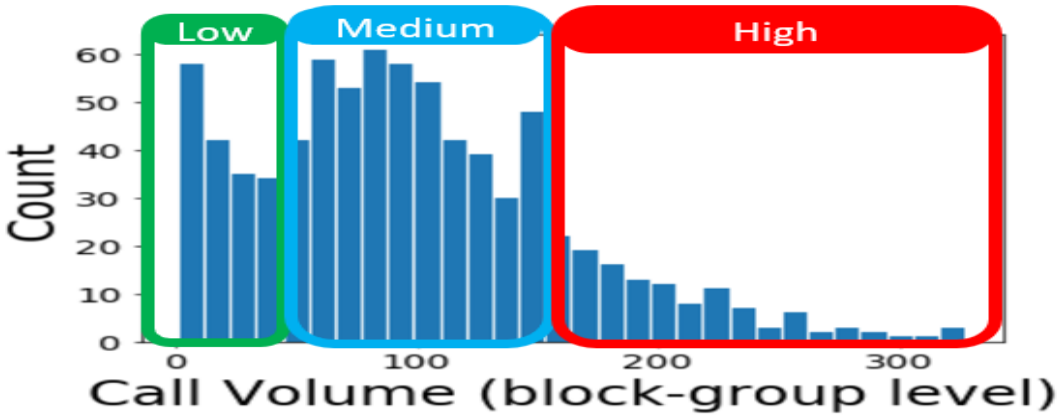


Figure 4.6: Categorical assignment on top of the histogram of service request volumes aggregated by the block group level in Miami-Dade County for 2013. The categories were labeled as low, medium, and high call volumes. The number of bins used was 30, and the records having a volume of more than 300 were discarded.

were two undirected edges between the target and independent variables, one between the Medium Volume and Female population node, and another between Low volume and Owner-occupied units. There were a total of 3 directed edges from the target variables to independent variables. These edges do not support our intuition since, in general, we do not expect an edge from the target variables to the independent variables. These spurious relationships may be caused by the presence of latent variables (confounders) as described in Section 4.5. A confounder is a variable that is either not measured or not used in the analysis, but causally impacts the two variables that are connected by an edge. Knowledge of confounders can help correct the dependencies between two variables, but are often a challenge to identify and measure. Since call volumes are considered as outcome variables, the following edges are potentially spurious: Low Volume $\rightarrow$ Single Unit; High Volume $\rightarrow$ Single Unit; and High Volume $\rightarrow$ Black Population.

Next, we narrowed down our analysis (Fig. 4.8) to only include the largest request type (i.e., Bulky trash request), since this category of request is dominant (501,972 out of 860,254, 58% of the total requests) in the data. As before, the nodes representing call

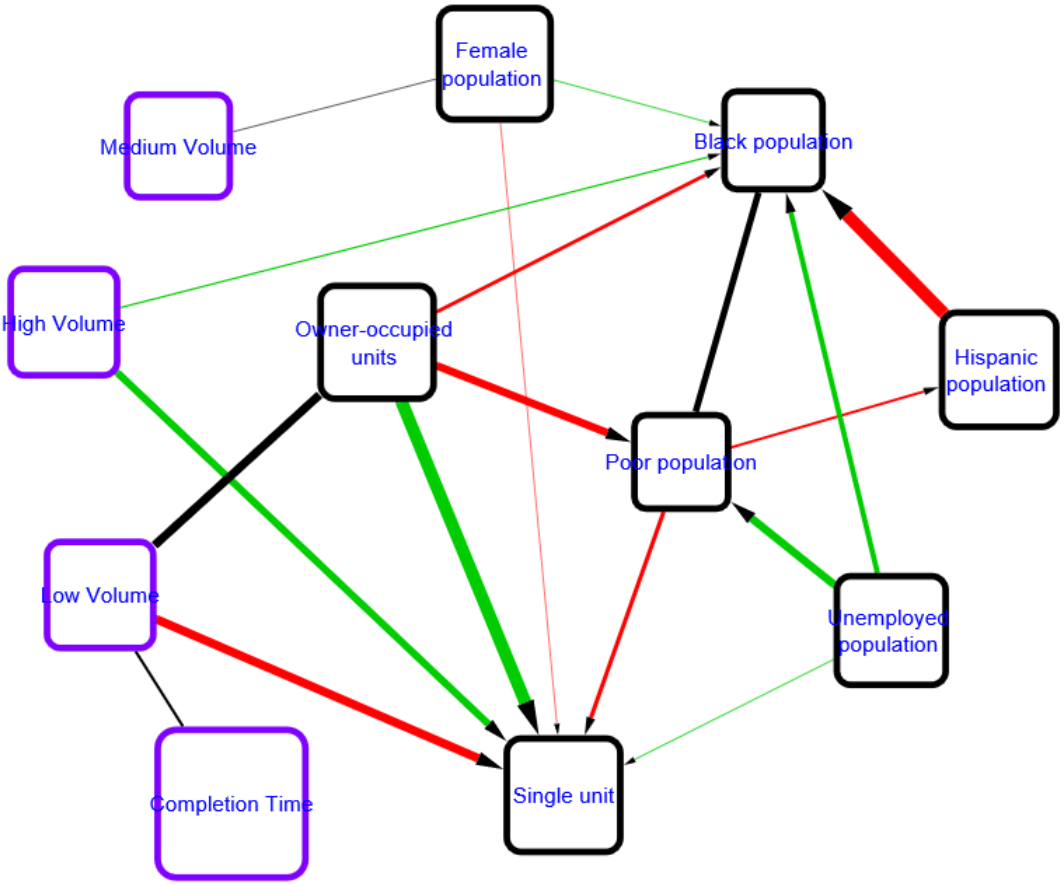


Figure 4.7: Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using all requests. The variables are aggregated across the block group. The network shows no incoming edges for the target variables (i.e., Low Volume, Medium Volume, High Volume, and Completion Time), suggesting no socioeconomic bias regarding citizen engagement and government responsiveness.

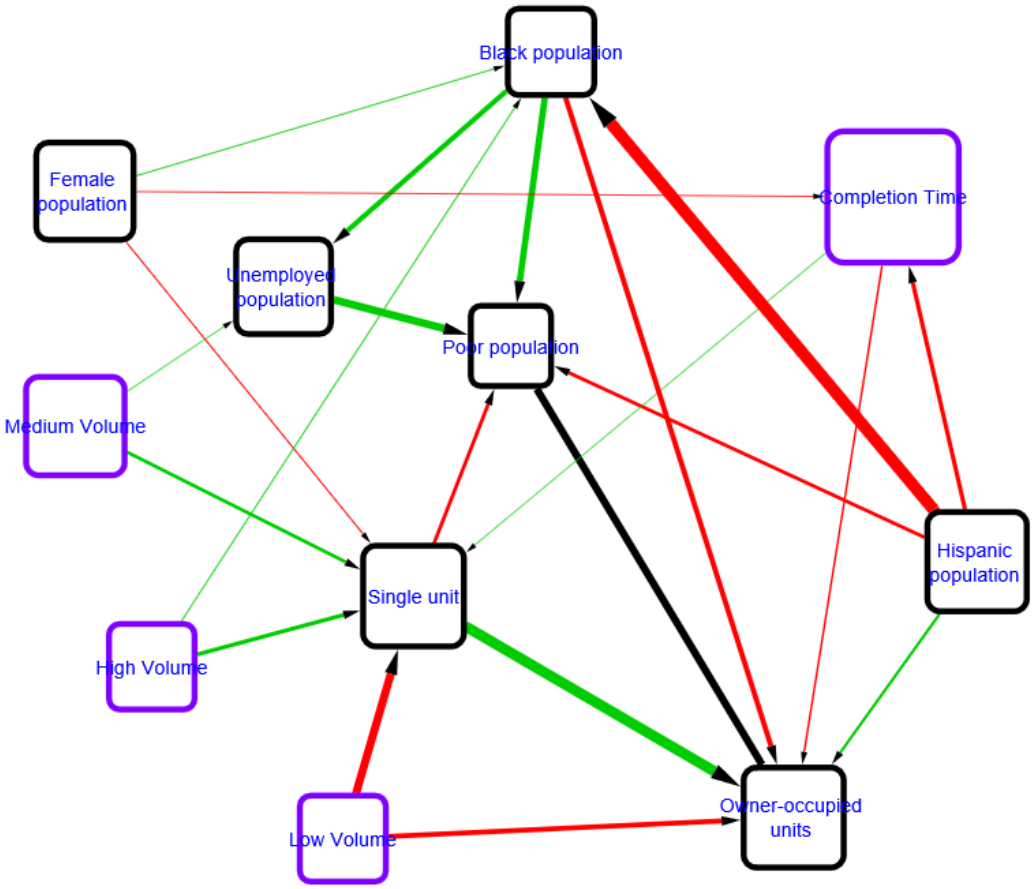


Figure 4.8: Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using the largest request type (bulky trash pickup) only. The variables were aggregated across the block group. The network showed no incoming edges for the target variables (i.e., Low Volume, Medium Volume, High Volume, and Completion Time), suggesting no socioeconomic bias regarding citizen engagement and government responsiveness.

volumes have no incoming edges suggesting that none of the independent variables affect the volume. Completion time has two incoming edges, one from a gender node (Female population) and another from an ethnicity node (Hispanic population). Both edges have negative Pearson correlation coefficients. Inspecting the inferred regression formula at the nodes suggests that the weights of these two edges are relatively low compared to the others in the network. There is one undirected edge connecting two independent variables, i.e., Poor population and Owner-occupied units, and which cannot be supported by intuition. There are a total of 8 directed edges from the target variables to independent variables that are also likely to be spurious. These edges include the following: Low Volume $\rightarrow$ Single Unit; Low Volume $\rightarrow$ Owner-occupied units; High Volume $\rightarrow$ Single Unit; High Volume $\rightarrow$ Black population; Medium Volume $\rightarrow$ Single Unit; Medium Volume $\rightarrow$ Unemployed population. Completion time $\rightarrow$ Single unit; and Completion time $\rightarrow$ Owner-occupied units.

Some edges are intuitive in the network, i.e., Unemployed population $\rightarrow$ Poor population, Single unit $\rightarrow$ Owner-occupied units, Single unit $\rightarrow$ Poor population. It is well known that unemployment contributes to the increase in poverty. The other two edges suggest that the single unit is a common cause in this network, influencing both owner-occupied units and poverty. Usually, the single units are owner-occupied, and it is less likely the owner will suffer from poverty. Also, the edge from Female to Black population is supported by the literature [Female-headed black family statistics in the US].

Census tract:

Next, we generated the sBNs for the Census tract level. We combined the census data with the aggregated completion time and categorized requested volume on the Census tract level for this task. First, we analyzed the dataset of all requests. The network (Fig. 4.9) generated from this set had comparatively fewer number of edges compared to the

network from the block level. As we aggregate larger geographic regions, the data exhibits less diversity in terms of community characteristics. This may explain why we find fewer interactions among the variables.

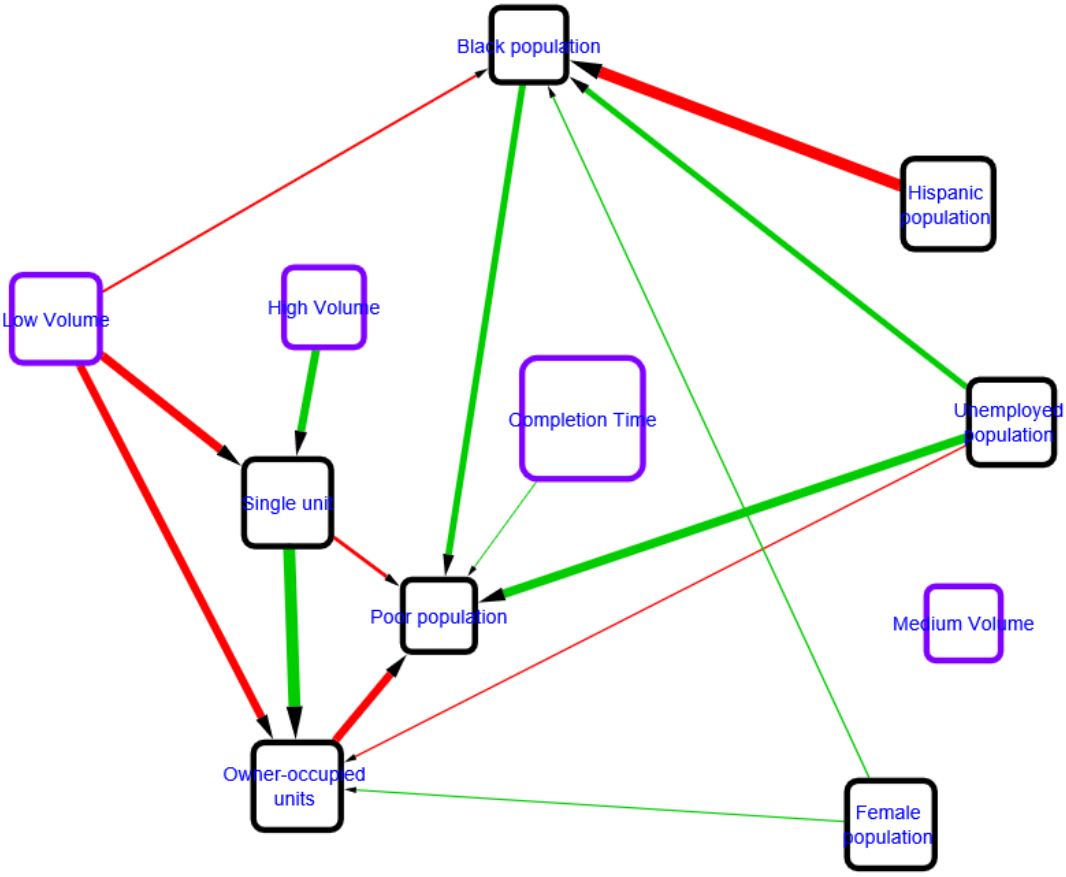


Figure 4.9: Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using all requests. The variables are aggregated across the census tracts. The network shows no incoming edges for the target variables (i.e., Low Volume, Medium Volume, High Volume, and Completion Time), suggesting no socioeconomic bias regarding citizen engagement and government responsiveness.

The network has no undirected edges. The target nodes have no incoming edges. There are a total of 5 directed edges from the target variables to independent variables that are also likely to be spurious. These edges include the following: Low Volume $\rightarrow$ Single Unit;

Low Volume → Owner-occupied units; Low Volume → Black population; High Volume → Single Unit; and Completion time → Poor population.

As before, we also generated the sBNs for the largest request group only (Fig. 4.10). This network has one undirected edge between the Black and Unemployed population. The target nodes have no incoming edges. There are a total of 6 directed edges from the target variables to independent variables. These include: Low Volume → Single Unit; Low Volume → Hispanic population; Completion time → Hispanic population; Completion time → Owner-occupied units; High Volume → Single Unit; and Medium Volume → Single Unit.

4.4.3 Discussion

From the resulting networks, we observe that the potentially spurious edges Low Volume → Single Unit and High Volume → Single Unit are present in all the networks. The directions are also consistent. It is unusual for target variables to have such outgoing edges to other variables in the network. The algorithms generating these networks are based on heuristics and assumptions, which could lead to wrong inferences. One of the issues with algorithms based on the CI test is that an edge might appear between two variables if a variable (confounder) that affects both these variables is not included in the analysis. Therefore, there is a possibility that confounding variables (common causes) exist for this dataset. A spurious edge may also lead to spurious chains and other spurious substructures of importance.

v-structures

If the dataset has no hidden confounders, then we can be most confident about the directions of the edges in v-structures in the network. The directions of the rest of the edges are

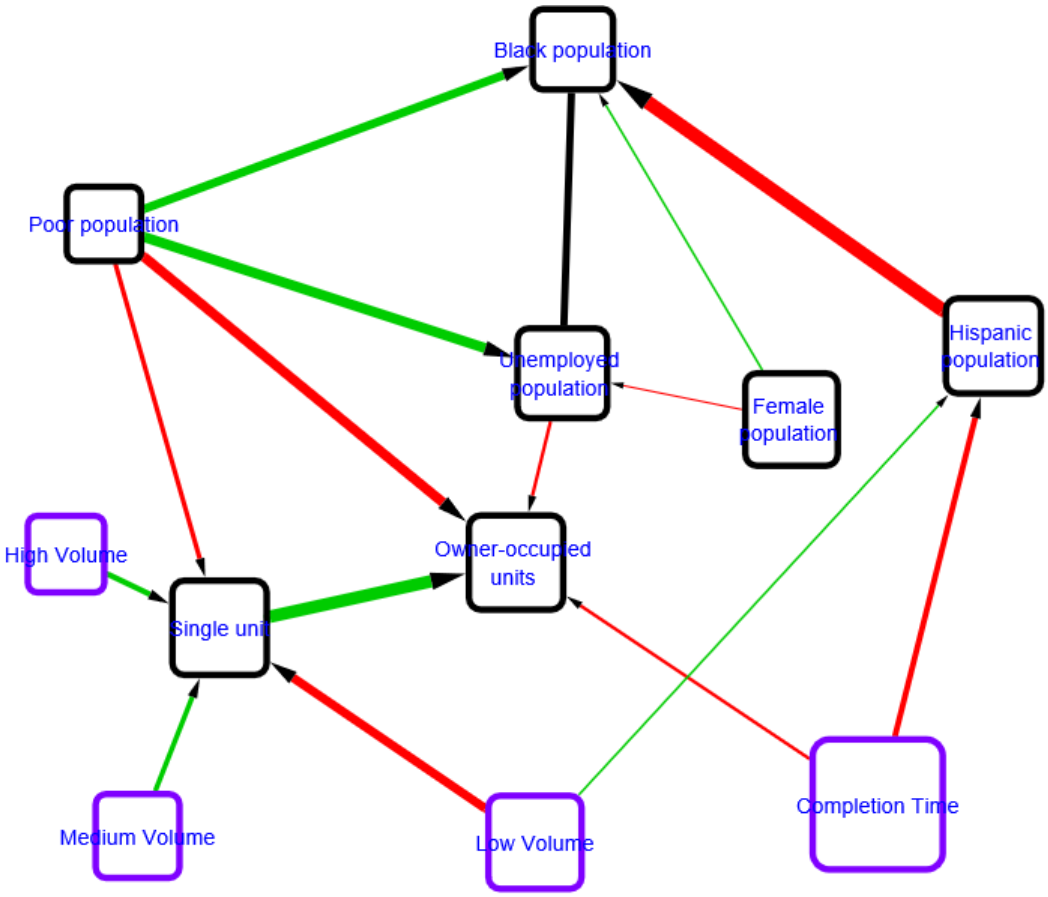


Figure 4.10: Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using the largest request type (Bulky trash pickup) only. The variables are aggregated across the census tracts. The network shows no incoming edges for the target variables (i.e., Low Volume, Medium Volume, High Volume, and Completion Time), suggesting no socioeconomic bias regarding citizen engagement and government responsiveness.

not uniquely determined by the CI tests as explained in Section 4.3. Although any two edges incoming into a node may appear to be a v-structure, they are labeled as v-structures only after they can be confirmed using the CI test. Our analyses identified three different v-structures, some of which appeared in more than one of these networks.

(a) The v-structure, Hispanic population $\rightarrow$ Completion Time $\leftarrow$ Female population, is an excellent example of a target variable identified as the "common" effect of two independent variables. Both the Hispanic and Female population is inferred to affect Completion time. This structure is only present in the block group data with bulky trash pickup requests. The correlation is negative for both the edges which indicates that the Completion Time decreases with an increase in the Hispanic and Female populations. Neither demographic group can be considered as "minority" in the context of Miami-Dade County (Table 4.2).

(b) The v-structure, Hispanic population $\rightarrow$ Black population $\leftarrow$ Female population, suggests that a rise in the Hispanic population causes the size of the Black population to decrease. This is consistent with the data that the Hispanic population in Miami-Dade County is predominantly white; The other edge suggests that a rise in the female population results in an increase in the percentage of the Black population. This is consistent with published results from the literature that suggest that *female-Headed* black families has seen an increase in the USA over the years [Female-headed black family statistics in the US]. This structure was inferred in all structures, using both block and Census tract data.

(c) The v-structure, Unemployed population $\rightarrow$ Poor population $\leftarrow$ Single unit, suggests that the size of the unemployed population results in a rise in the indigent population, which makes intuitive sense. In contrast, the percentage of the Single units in the community affects the Poor population percentage. It is reasonable that the

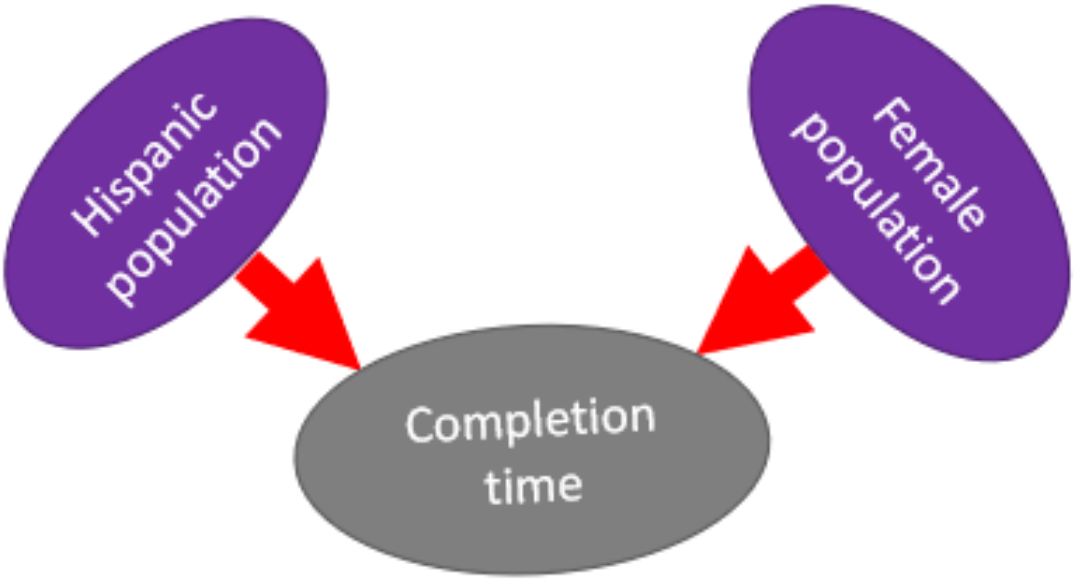


Figure 4.11: A v-structure involving only demographic and target variables (i.e., the percent of Hispanic population, female population, and average completion time in the community aggregated across the block group) appearing in the Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using the largest request type only (Bulky trash pickup).

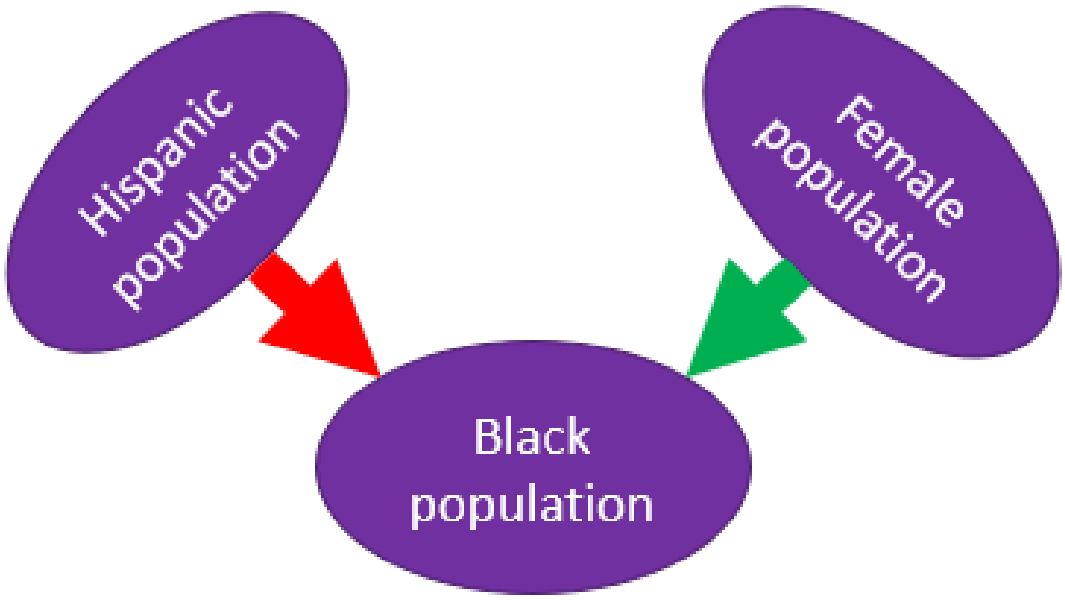


Figure 4.12: A v-structure involving only demographic and socioeconomic variables (i.e., the percent of Hispanic, female, and black population in the community aggregated across the block group) appearing in the Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using all requests.

correlation is negative, however the direction of the edge may be spurious. This v-structure is only inferred from the block group data with bulky trash pickup requests.

Analyzing calls of type "Complaints":

We also examined another type of request, namely complaints. "Complaints" usually take longer (8.5 days on average for block group level) than "Requests". The results from Figure 4.14 indicate that there is no effect of demographic or socioeconomic status on completion time for this dataset.

This network also has the v-structure, Hispanic population $\rightarrow$ Black population $\leftarrow$ Female population we identified in 4.12. The nodes representing the target variables are disconnected from all the others except the Low volume. Low volume is affected by both

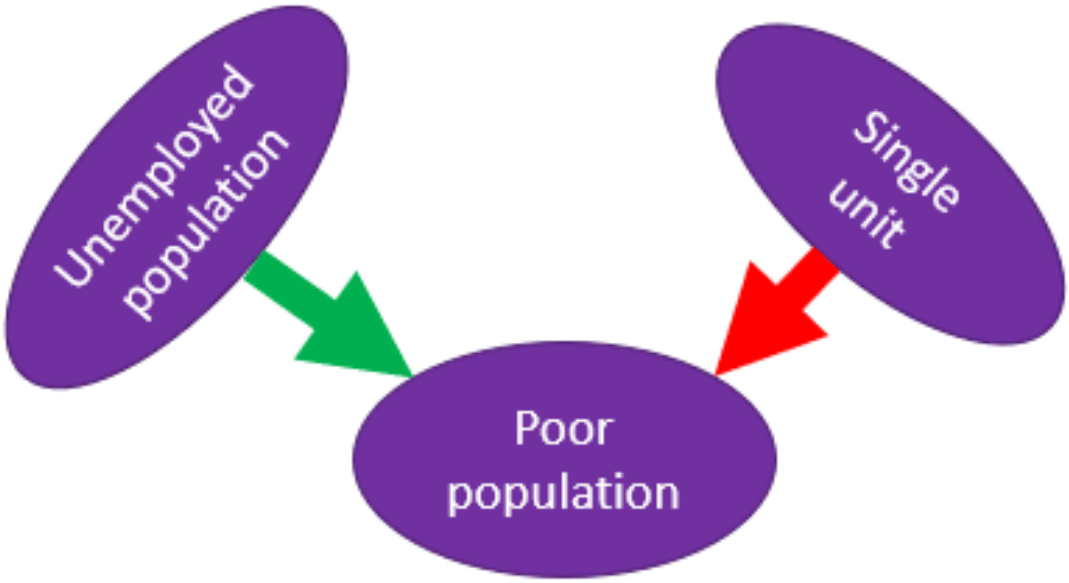


Figure 4.13: A v-structure involving only demographic and socioeconomic variables (i.e., the percent of unemployed population, single housing unit, and poor population in the community aggregated across the block group) appearing in the Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using the largest request type only (Bulky trash pickup).

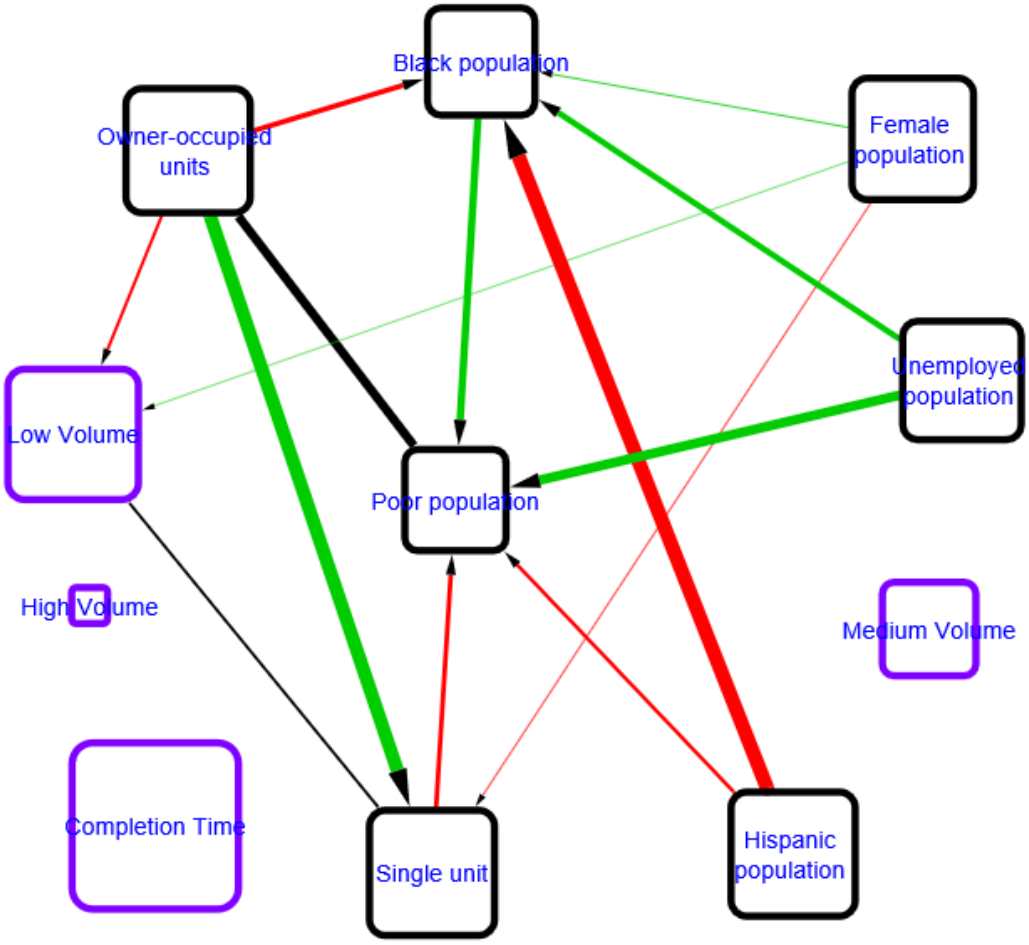


Figure 4.14: Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using all "complaints". The variables are aggregated across the census tracts. The network shows "Low Volume" affected by the percentage of female population and owner-occupied units. This suggests some socioeconomic bias regarding citizen engagement, but not strong enough as the weights of the edges for those are very low.

the Owner-occupied units and Female population. One of the findings is that completion time is not affected by any other variables indicating no bias induced by community characteristics.

4.5 Challenges with causal Bayesian networks

The main challenges in inferring causal relationships from observational data arise from the fact that there are several assumptions made in applying causal inferencing. Real-world observations may not always follow those assumptions; hence, it introduces challenges in applying the method successfully.

4.5.1 Missing and impure data

The process of data acquisition process often results in incomplete administrative datasets. Administrative datasets may contain missing data points, and may have recording errors. For example, in the 311 data, we found records whose request status had never been closed. Other records show the closing date recorded to be prior to the open date. Such inaccurate records (16.28% of the total) must either be manipulated or ignored, thus reducing the number of accurate observations available for analysis. When we excluded the inaccurate data points, the number of total observations decreased.

4.5.2 Inadequate data

The accuracy of AI models also depend on the quantity of available data. Large amounts of good data can help develop more robust models as they can better capture the underlying relationships under investigation. This requirement becomes higher for high-dimensional data.

Even though the total number of observations may be large for some 311 datasets, a single record is not associated with the features of the individual who requested the service; this is done to ensure anonymity in public datasets. As a result, we considered the community characteristics rather than metadata associated with individuals, obtained by aggregating the data on an appropriate geographic unit, such as a block group or a census tract. When we aggregated the data on a geographical unit, the resulting number of observations is reduced while the geographical location data gets coarser. For example, when we aggregated the Zip code level observations, the number of records becomes considerably smaller (5-fold reduction from the Census tract level). Inadequate data tends to produce a misleading model. Barring such fine-grained information from the 311 data, there is a greater possibility for biases in the data (e.g., by a small set of individuals making most of the requests). An alternative data collection approach that collects demographic data of each individual requester could add valuable richness to the analysis proposed here.

4.5.3 Latent confounders

Finally, there exist variables that are missing in the dataset, either because they were not included in the analysis, not measured, or because there was no known way to measure them. These confounders can impact the analyses by creating incomplete models, and leading to potentially incorrect inferences. These variables are called latent "confounders". Structure learning models are based on the assumption that all the independent variables affecting the target variables are present in the observation, which may not be true. In such cases, the model cannot discover the real cause-effect relationships accurately. It is often impossible to avoid the possibility of having unobserved variables in the real world because they are often unknown. When the data fails to capture the key factors of interest, the model will also be inadequate in explaining the findings. Our analysis discovered that

Low Volume → Single Unit and High Volume → Single Unit are recurring edges in many of the inferred networks. Intuitively, the requested volume should not cause the housing conditions to change. We can explain this edge with the help of confounding variables. We assume that a latent variable, not included in the network, affects both the requested volume and housing condition resulting in a directed edge between them. We used a limited number of demographic and socioeconomic information. Our analysis did not include any information regarding the department (a specific unit that handles particular types of requests), infrastructure, and resources. Excluding that information may result in an incomplete model since the available resources may impact the efficiency while also being correlated to the types of homes in that block.

Also, since the structure learning algorithms are based on heuristics, more than one structurally equivalent BN can be obtained from the same observations. Once the v-structures are identified, the structure learning algorithm's last step assigns the remaining directions based on some predefined rules. These rules may still leave some edges to be undirected. Also, the directions inferred may be inconclusive in some cases due to ambiguity in determining the causal chain and common cause structures. We tried to overcome this limitation with bootstrapping by generating several models, and assigning weights to each edge.

We applied causal inference resulting in causal Bayesian network models, which helps to determine the demographic and socioeconomic factors that have a causal impact on the target variables such as completion time and call request volumes. We concluded that the results do not support any claims of demographic and/or socioeconomic bias in providing non-emergency services to the residents of Miami-Dade County. The case study using data from just one city cannot ensure that data from other cities or municipalities will result in the same conclusion. However, the findings are consistent with extant research on 311 data from other cities, as will be discussed in Chapter 5. More importantly, this chapter

provided a framework to apply causal inference on 311 datasets, which can be readily extended to data from other cities or regions. Finally, we also provided a discussion on the challenges in applying the causal approach to this type of dataset.

CHAPTER 5

COMPARATIVE ANALYSIS OF 311 DATA FOR DIFFERENT CITIES

5.1 Motivation

Citizens and government have been able to work together to improve services and citizen satisfaction because of advancements in e-governance. The 311 systems deal with non-emergency service requests from the local community and complement the 911 emergency services. A resident can report a problem, a complaint, or a request for local government services. Examples of reports include a tree branch blocking a sidewalk, a stoplight malfunction, and garbage pickup requests. Depending on the city, citizens report by calling, visiting the website, or by using a smartphone-based application. Although not every city offers the services, it has grown in popularity over time, and more communities are embracing the 311 concept, and has been adopted by around 100 cities [Thomas, 2013]. Another welcome trend is referred to as "Open 311", wherein cities have begun to make the call data publicly available, thus making the 311 system more transparent, accountable, effective, and efficient for the public. Understanding the citizens' needs and demands and the provision of appropriate services in response to such demands are of interest for evaluating the responsiveness and effectiveness of the local government. These trends have paved the way for cities and municipalities to adopt new technologies to transparently collaborate with citizens and better manage their resources.

The main objective of the previous chapter was to provide a causal framework to evaluate the local government's response in providing non-emergency services [Yusuf et al., 2021a]. The framework helped us to answer the question: is there any indication of bias against a racial, ethnic, or under-privileged population in the community? A primary motivation for studying Bayesian networks (BNs) comes from the fact that most statistical modeling approaches can demonstrate correlations, but cannot establish causation. Instead

of merely looking through the lens of correlation, we seek answers with the power of causal inference.

BNs allow us to process a data set and represent probabilistic relationships between the variables of the data set as a directed network and a set of conditional probability tables. BNs provide a graphical depiction that is easy to understand and interpret. BNs capture more complex and informative relationships between variables than traditional models. Causal BNs allow us to model causal relationships between variables and provide a framework for powerful predictions of situations not represented in the data. BNs allow us to study *interventions* and *counterfactuals*. In short, they help us to do a wide variety of causal inferencing and have a wide range of application domains. In the public policy domain, they can help us understand the causal structure of our data, evaluate probabilities of events, assess interventions (i.e., policies) before implementing them, and even contemplate counterfactuals. Therefore, we applied the causal BN approach to the 311 data to shed some light on the questions regarding the equitable response of local governments. In Chapter 4, we applied causality to the 311 data from Miami-Dade County. In this chapter, we extended the framework to data from two more cities – New York City and San Fransisco. In the process, we investigate several challenges related to inadequate data, confounders, transfer learning, and the comparison of causal BNs.

5.2 Causal Bayesian networks

As described in Section 4.3, BNs are a structured and graphical representation of probabilistic relationships between several random variables. Arcs encode conditional dependence. BNs also satisfy a technical condition called the Markov condition. Causal networks are special networks that connect two variables if and only if they have a direct causal relationship. Causal networks are known to be BNs. Thus the theory of BNs helps us

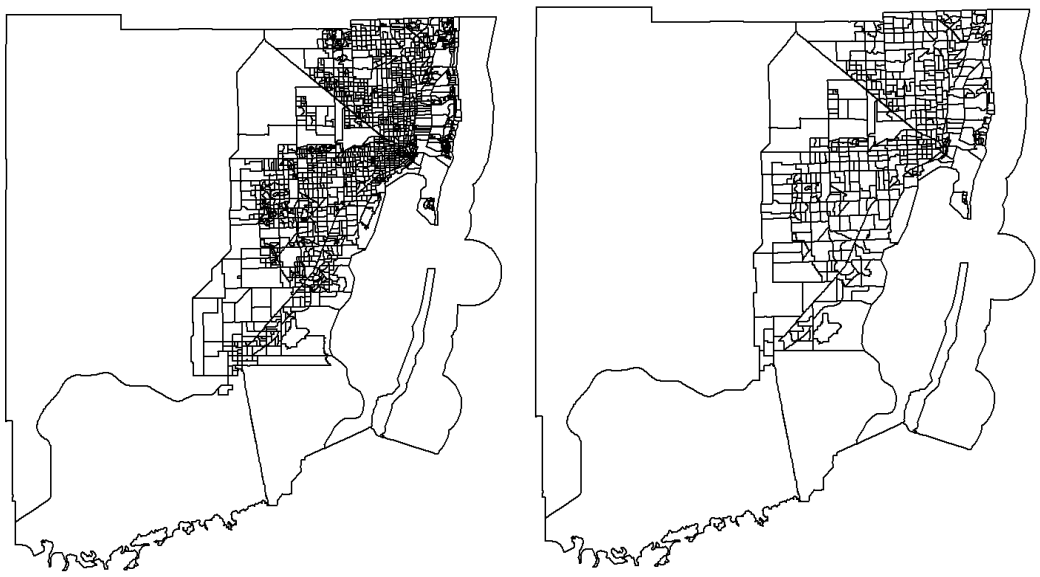
study causal networks and allows for causal inferencing to be performed. However, it has several limitations while dealing with observational data. We will discuss some of these limitations and ways to overcome those in the context of the 311 dataset.

5.2.1 Inadequate data

As discussed in 4.5.2, the accuracy of learned models is influenced by both the quantity and quality of the available observations. Not only do they have to be reliably measured (i.e., accurate), but they must also reflect the large number of combinatorial possibilities in the data. Large amounts of good data can enhance the quality of the learned models by capturing the underlying relationships. If the data is not adequate, the network learned might be misleading.

Even though we have access to data on a large number of 311 calls, the data does not include important characteristics of the individuals making the request (gender, race, ethnicity, economic status, household characteristics, etc.) in the interest of anonymity. As a result, rather than considering metadata connected with individuals, we evaluated community characteristics generated by aggregating data on an appropriate geographic unit, such as a block group or a census tract. This weakens the causal approach, which could otherwise have arrived at powerful relationships.

Figure 5.1 shows that the block group is a smaller geographic unit than the census tract and is associated with more granular community characteristics compared to the census tract level. As a result, we conducted all our subsequent experiments on the block group level to retain more observations. Also, it explains more variability as the block level is more granular compared to the census tract level.



(a) Block groups. (b) Census tracts.

Figure 5.1: Maps of Miami-Dade County divided into (a) block groups, and (b) census tracts.

5.2.2 Potential confounders

As discussed in 4.5.3, one of the challenges of learning BNS from observational data is the possiblity of "latent confounders". Confounding factors are missing from the dataset and result in incomplete models and potentially inaccurate inferences, either because they were measured but not used, not measured, or there was no known way to quantify them. As an example, for the 311 data analysis, we have not included the time information. As another example, analysis of the 311 data set for Miami-Dade County shows that call volume is correlated with completion time (Fig. 4.7). However, proving whether or not this relationship is causal requires much more than a correlation computation. Conditional analysis with one or more additional variables may be needed. It turns out that when conditioned on the day of the week or month of the year, call volume and completion time is (conditionally) independent, thus showing that day/month variables are confounders for the relationship between call volume and task completion time. Fortunately, the 311 data

set did provide temporal information, thus allowing us to include this confounder. Other potential confounders that are not included in the data set are the number of service crews available and the budget of the department handling the requests. A well resourced 311 center can maintain task completion time even if it spikes for other reasons. At the same time, such a 311 center can also encourage citizens to be more active in posting requests.

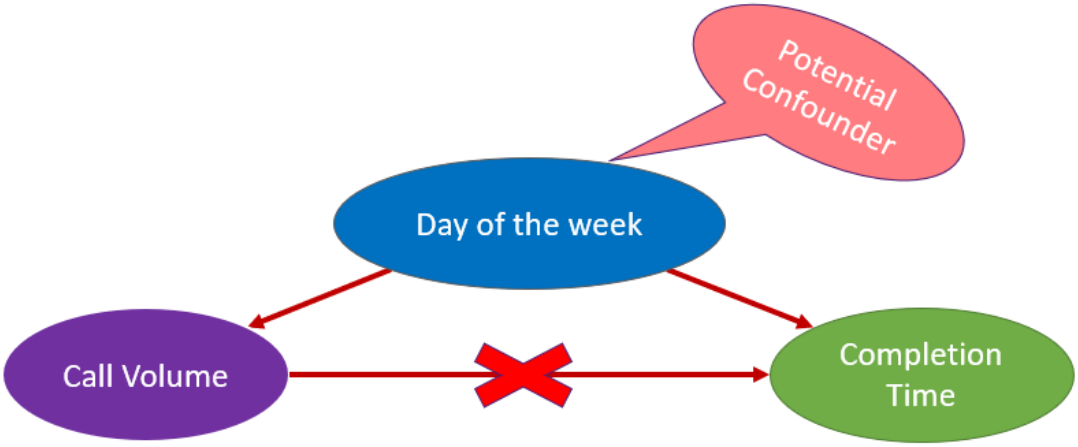


Figure 5.2: Latent confounder – the day of the week when the request was made is known to affect the call volume as well as the completion time, and is latent confounder that limits the accuracy of inferences from the 311 data analysis.

5.2.3 Integrating prior knowledge and adjusting the parameters

As mentioned above, structure learning allows us to determine which arcs are present in the causal network, as evidenced by the data. In Chapter 4, we learned the causal network using a completely data-driven approach. In many applications, we have domain knowledge that helps us determine which arcs should be disallowed. For our case study, we experimented with applying some prior knowledge.

Bayesian networks allow the integration of such prior knowledge in the network. *Whitelists* and *blacklists* are one way to accomplish this. In the `bnlearn` package, both

are implemented as follows: Whitelisted arcs are always retained, while blacklisted arcs are always excluded from the network. For the 311 data set, we assume that the demographic variables are not directly causally affected by the measured 311 variables. Therefore, all edges leading from target variables (completion time and volume requests) to the demographic and socioeconomic variables were put on the blacklist for the network. Also, the CI test in 'bnlearn' package has an adjusting parameter *alpha*, denoted by α, which is defined as the type *I* error threshold for the CI test and can improve the confidence of the arcs in the network. We increased the confidence threshold from 95% to 99% for the CI test, resulting in a network where we have more confidence in the edges. Also, we generated 200 networks from the same observations as the learned structures can be different at each run due to randomization in the algorithm (Pc-Stable) used. We then used the 200 resulting networks to generate a final "consensus" network, where the edges appear with their thickness proportional to the *bootstrap strength score*. The bootstrap strength score is the count of the fraction of times the edge appears in the network out of the $N = 200$ runs.

5.3 Improved casual Signed Bayesian network for Miami-Dade County

As mentioned in Section 4.4.1, we collected and processed data from 311 requests and census data. Table 5.4 and Table 5.5 provide the details of all the columns available from 311 open data hub. Except for the ticket creation/update date, time, and location, all the columns are either descriptive or categorical. Therefore, we only used the time and location information to derive the *Completion time* and *Block group id* from the geolocation data. We also included the Service request type for the filtering purpose. The main reason for excluding the categorical variable is that the number of discrete types in each category column is vast (i.e., more than 200 for service request type). "One hot" encodings of these

categorical variables can result in a very high-dimensional dataset, which is not feasible for structure learning. As a result, we excluded the categorical and descriptive columns from our analysis. Location values were made discrete by assigning the same location value to all GPS coordinates from the same block group level of Miami-Dade County. Demographic data include the percentages of Black, Hispanic, and Female populations from the ACS Census data for each block group. The percentages of single-family units and owner-occupied units are all housing features. Housing characteristics can be used to regulate community "quality" or service demands and capture differences in housing stock across block groups. Unemployment rates and poverty rates are all economic characteristics of a community. In addition, to account for possible capacity differences between neighborhoods, the total number of service requests (call volume) received within each block group was also considered. Table 5.1 summarizes the mean and standard deviation of the variables under consideration in the dataset.

Table 5.1: Mean and standard deviation are shown for the variables used in the data set. Data is from Miami-Dade County obtained from Census (ACS) data and derived from 311 data. The volume (total) and completion time (average) are aggregated for all requests from the same block group

Variable name	**Block group**	
	Mean	**SD**
Volume	4.38	4.48
Completion Time	13.8	12.5
Owner-occupied units	0.57	0.26
One unit	0.59	0.35
Female population	0.51	0.06
Black population	0.20	0.28
Hispanic population	0.62	0.28
Unemployed population	0.42	0.11
Poor population	0.15	0.12

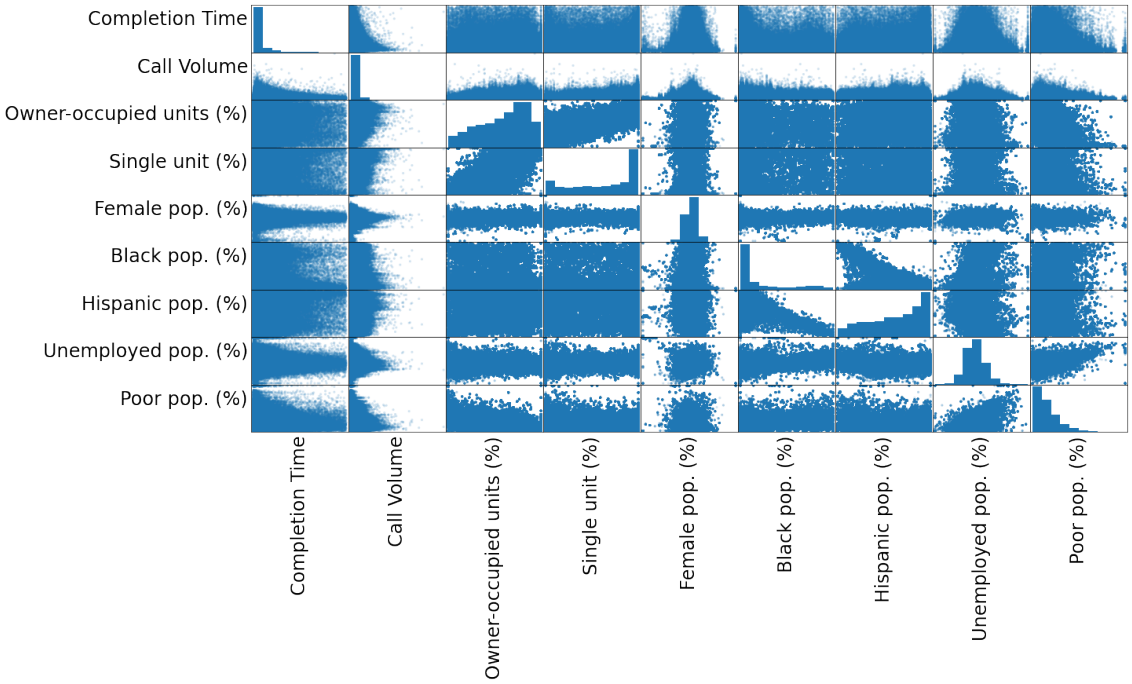


Figure 5.3: Matrix of scatter plots of Census (from ACS data) and derived (from 311 data) variables for Miami-Dade County. The target variables, volume (total) and completion time (average), were aggregated by the block group, day of the week, and month of the year.

In Fig. 5.3, we provide a matrix of scatter plots of all pairs of variables considered for analysis, depicting the pairwise relationships between them. The plots along the diagonal are histograms of all the variables of interest. We observe that some variables, i.e., 'Owner-occupied units', 'Female population' and 'Unemployed population', exhibit a normal distribution. The derived variables from 311 data and some demographic and socioeconomic variables (i.e., 'Black population,' 'Poor population') show right-skewed long-tail distribution. On the contrary, 'Single housing units' and 'Hispanic Population' in M-DC show left-skewed long-tail distribution. The other plots in the matrix show the correlation between a pair of variables. From the plots, we observe some patterns, i.e., 'Owner-occupied units' and 'Single housing units' are positively correlated, 'Owner-occupied units' and 'Poor population shows a negative correlation, 'Unemployed' and 'Poor population' have a positive correlation.

We fed the preprocessed data to the `bnlearn` R package [Scutari, 2009] and utilized the "PC-stable" algorithm, a constraint-based structure learning method. We augmented the edges of the generated network with colors based on the Pearson correlation coefficient as suggested by [Fernandez et al., 2015, Sazal et al., 2020a]. The weights of the edges are equal to the bootstrap value (equal to the percentage of the number of times the edge appears in the network). The resulting network is referred to as the *Signed Bayesian network* (sBN) [Sazal et al., 2020a].

Figure 5.4 shows the generated network. The target variables, i.e., completion time, low volume, medium volume, have incoming edges. However, the weights of those edges are very low compared to the other edges in the network, suggesting that the influence of the variables on the outcome variables (volume or completion time) are not statistically significant. Now focusing on interpretability of this particular AI technique, we begin with the edges in the network (Fig. 5.4). From the network, we can see red outgoing arcs from "Hispanic population" to Population living under the poverty line ("Poor population")

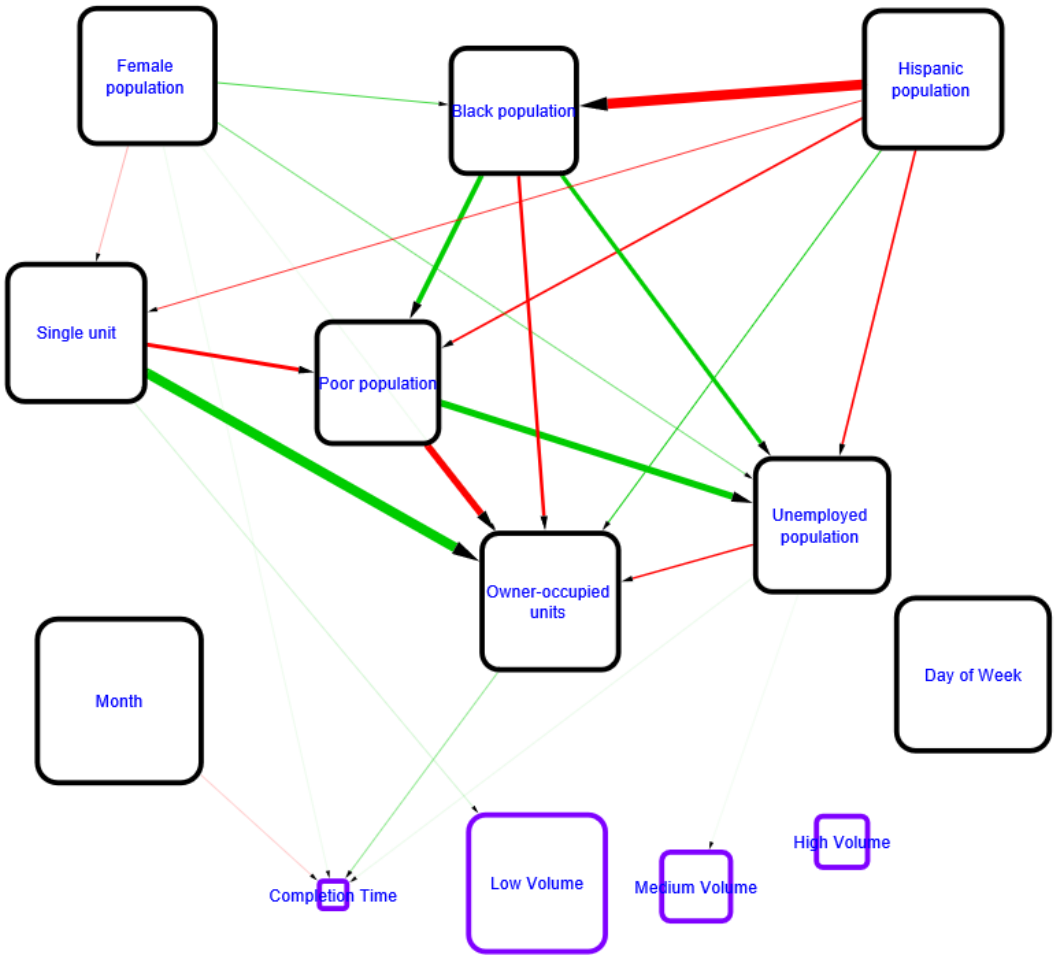


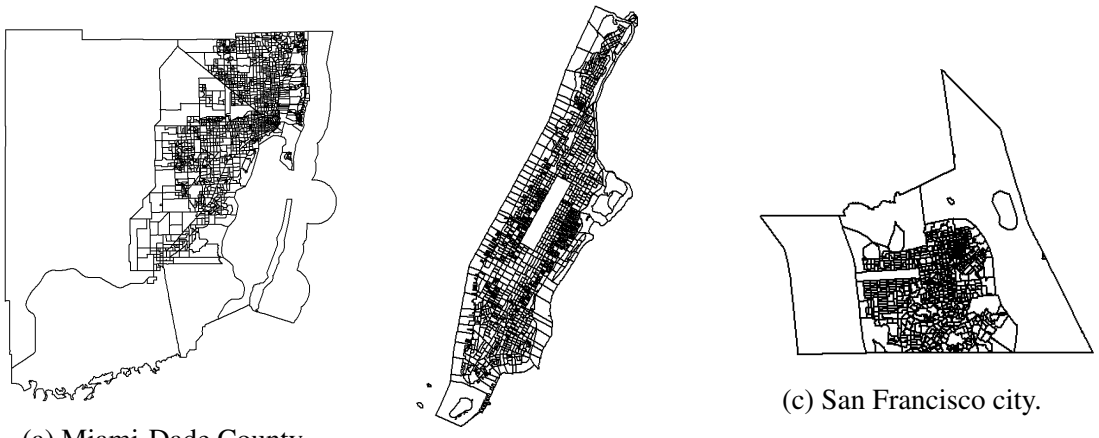
Figure 5.4: Signed Bayesian network generated from Miami-Dade County 311 datasets for the years ranging from 2013 to 2019 using all requests. The variables are aggregated by the block group, day of the week, and month of the year. The network suggests small traces of socioeconomic bias (with low statistical significance) with regard to government responsiveness, as indicated by the edges from the demographic and socioeconomic variables to the target variables like "Completion time", "Low Volume", and "Medium Volume".

and the “Unemployed population”. This is supported by the fact that in Miami-Dade, a substantial fraction of the Hispanic communities have high socioeconomic status (SES) and live well above the poverty line. Also, “Single unit” houses are usually “Owner-occupied” and the edge connecting them indicates a strong positive correlation, including a positive correlation with the Hispanic community. Also, we investigate the causal effect of demographics and socioeconomic variables on the target variable. It can be concluded from the generated network, that “Completion time” and call volumes (represented by “Low Volume”, “Medium Volume”, and “High Volume”) are affected by different variables, but these edges do not have strong bootstrap support, suggesting low confidence on a purported causal relationship. In contrast, the edges from “Single unit” to “Owner-occupied units” and from “Poor population to “Unemployed population” and “Owner-occupied units” are quite thick, suggesting strong bootstrap support for these causal relationships.

5.4 Extension of the framework to New York City and San Francisco

Finally, we extended our experiment to include two major cities, i.e., New York City and San Francisco. Both were early adopters of 311 and have data available from 2013 to 2019. Given that as we go up in the hierarchy of geographical units, the number of aggregated observations becomes less, we conducted all our experiments on a block group level. The block group level is the smallest unit and results in a higher number of aggregated records than either the census tracts or zip codes. New York City (NYC) has more than 6,000 block groups, and San Francisco (SF) has 2,332 block groups 5.5. The call volumes of both the cities are larger than that of Miami-Dade County. Furthermore, both cities have 311 data available from 2013 to 2019. They are also geographically distant from each other. Thus, the analysis of this data adds geographical diversity to the analysis. Finally, as

with Miami-Dade, NYC and SF are also racially and ethnically diverse with distinctive neighborhoods, adding additional examples to the study of inequity and diversity.



(a) Miami-Dade County.

(b) New York city.

(c) San Francisco city.

Figure 5.5: Maps of the three cities considered for our comparative analysis: (a) New York City, (b) Miami-Dade County, and (c) San Francisco. The highlighted polygons represent the block groups in each region.

We downloaded the publicly available 311 call data for New York City [New York City open data], and San Francisco [San Francisco 311 data]. The total number of records is approximately 4.8 and 21 million for New York City and San Francisco, respectively. San Francisco has data available from 2008, whereas New York City provides the data from 2010. However, ACS census data does not provide the estimate for all the considered census variables until 2012. As a result, we only considered the dataset from 2013 to 2019 for these cities, thus matching the period analyzed for Miami-Dade County. The service request volume plot aggregated by year can be found in Fig. 5.6

The New York City 311 data set has 41 columns (Tab. 5.4 and Tab. 5.5 from the appendix) describing the complaint types, the department that handles the complaint, opening date for the complaint (i.e., time complaint was lodged), closing date for the complaint (i.e., time the complaint was fully serviced), location of request, and more . The San Francisco 311 data set has 20 columns (Tab. 5.4 and Tab. 5.5), including the request

(a) New York city.

(b) San Francisco city.

Figure 5.6: Total number of service requests made to the 311 call centers by the local residents aggregated by year ranging from 2013 to 2019 in two different cities.

type, opening and closing time for the request, and the location (i.e., latitude and longitude) of the request. Following the same logic, we only considered the date, time, and location information from these cities and excluded the descriptive and categorical variables. As a result, we calculated the average completion time and total request volumes for both the cities aggregated by day of the week, the month of the year and the block group. Table 5.2 summarizes the important variables and basic statistics of both the 311 and census data variables.

Table 5.2: Relevant target and independent variables used from the 311 data sets of New York City and San Francisco, and their summary statistics.

Variable name	**NYC Block group**		**SF Block group**	
	Mean	**SD**	**Mean**	**SD**
Volume	5.51	1.42	8.25	11.43
Completion Time	165.54	1.06	21.68	63.97
Owner-occupied units	0.38	0.26	0.42	0.23
One unit	0.16	0.23	0.28	0.24
Female population	0.51	0.07	0.49	0.06
Black population	0.20	0.28	0.05	0.09
Hispanic population	0.23	0.22	0.14	0.13
Unemployed population	0.41	0.12	0.33	0.11
Poor population [Def. by ACS]	0.14	0.13	0.10	0.08

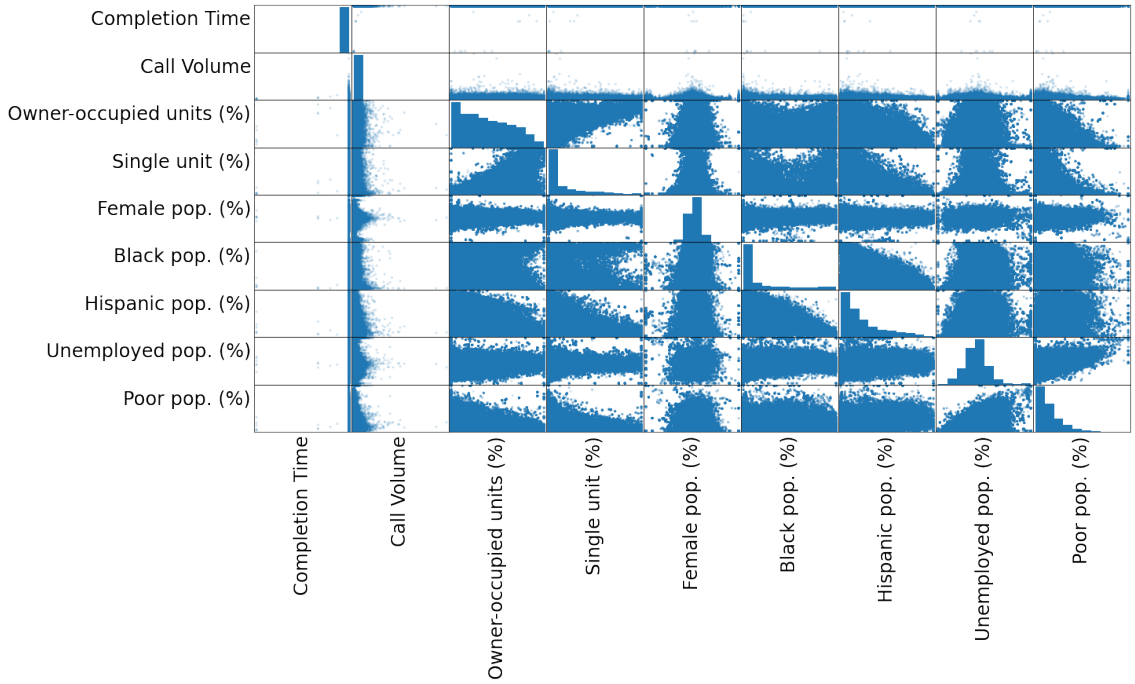


Figure 5.7: Matrix of scatter plots of Census (from ACS data) and derived (from 311 data) variables for New York City. The target variables, volume (total) and completion time (average) were aggregated by the block group, day of the week, and month of the year.

Figures 5.7 and 5.8 show a matrix of scatter plots of all the considered variables. We observe that some variables, i.e., 'Female population' and 'Unemployed population,' exhibit a normal distribution in NYC and SF (as in M-DC). However, 'Owner-occupied units' have right-skewed distribution in NYC, unlike in M-DC and SF. All the other variables exhibit right-skewed long-tail distribution in both NYC and SF. Also, 'Owner-occupied units' and 'Single housing units' are positively correlated, 'Owner-occupied units' and 'Poor population' show a negative correlation, 'Unemployed' and 'Poor population' have positive correlation in M-DC.

Figures 5.9 and 5.10 represent the generated signed Bayesian network for New York city and San Francisco. The target variable, completion time, has an incoming edge from the temporal variable, month. However, the bootstrap support for this edge is very low, suggesting that the causal influence on the target variables is negligible. There was little

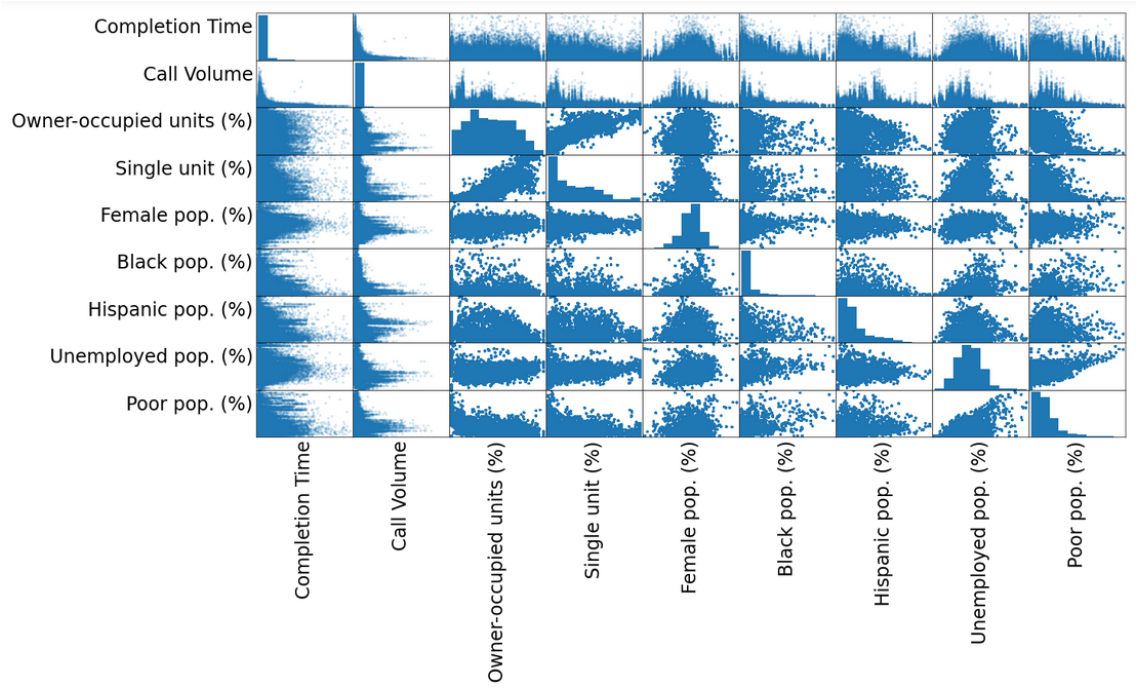


Figure 5.8: Matrix of scatter plots of Census (from ACS data) and derived (from 311 data) variables for San Francisco. The target variables, volume (total) and completion time (average) were aggregated by the block group, day of the week, and month of the year.

evidence in these studies to imply that systematic demand for 311 differed across a range of socioeconomic and racial factors. Also, the findings suggest that there are no systematic biases in how local governments deliver services to communities of color or those with lower socioeconomic status in the three cities under study.

In NYC and SF, the majority of Hispanic communities have low SES, unlike in M-DC. However, owner-occupied units are usually single units in NYC and SF, which is similar to M-DC as we see from the correlation plots that single units and owner-occupancy are positively correlated. Also, the month of the year the request was made has a weak effect on the service completion time in all the cities. Cold winters in NYC and the hurricane season in M-DC could account for this weak causal effect.

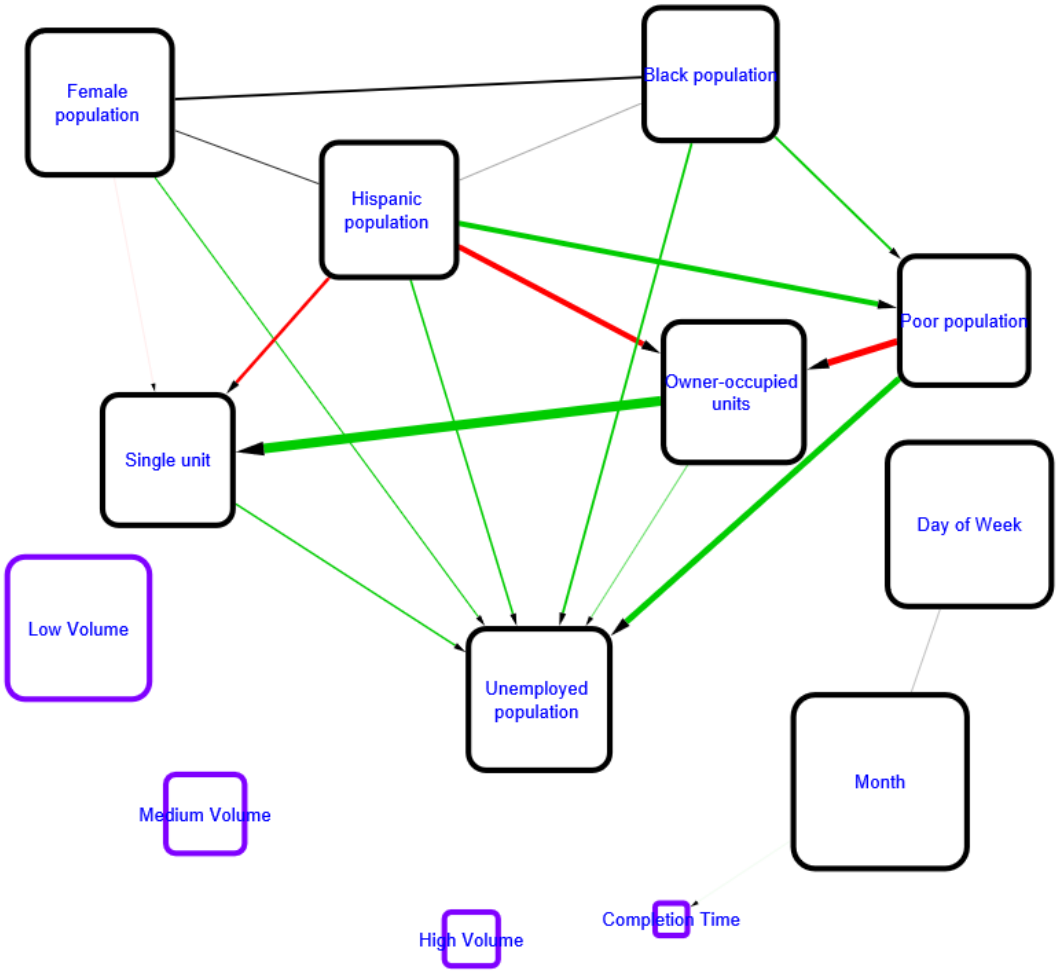


Figure 5.9: Signed Bayesian network generated from New York city 311 datasets for the years ranging from 2013 to 2019 using all requests. The variables are aggregated by the block group, day of the week, and month of the year. The network suggests one weak temporal bias (target variable, i.e., completion time affected by the temporal variable) with regard to government responsiveness, but with very low statistical significance.

5.5 Performance analysis

Many disciplines of research are interested in understanding cause-effect relationships between observed variables. Typically, experimental intervention (randomized controlled trials) is employed to validate these connections, which are often hypothesized only after gaining a lot of domain-specific knowledge and deep intuition. Experiments are, however,

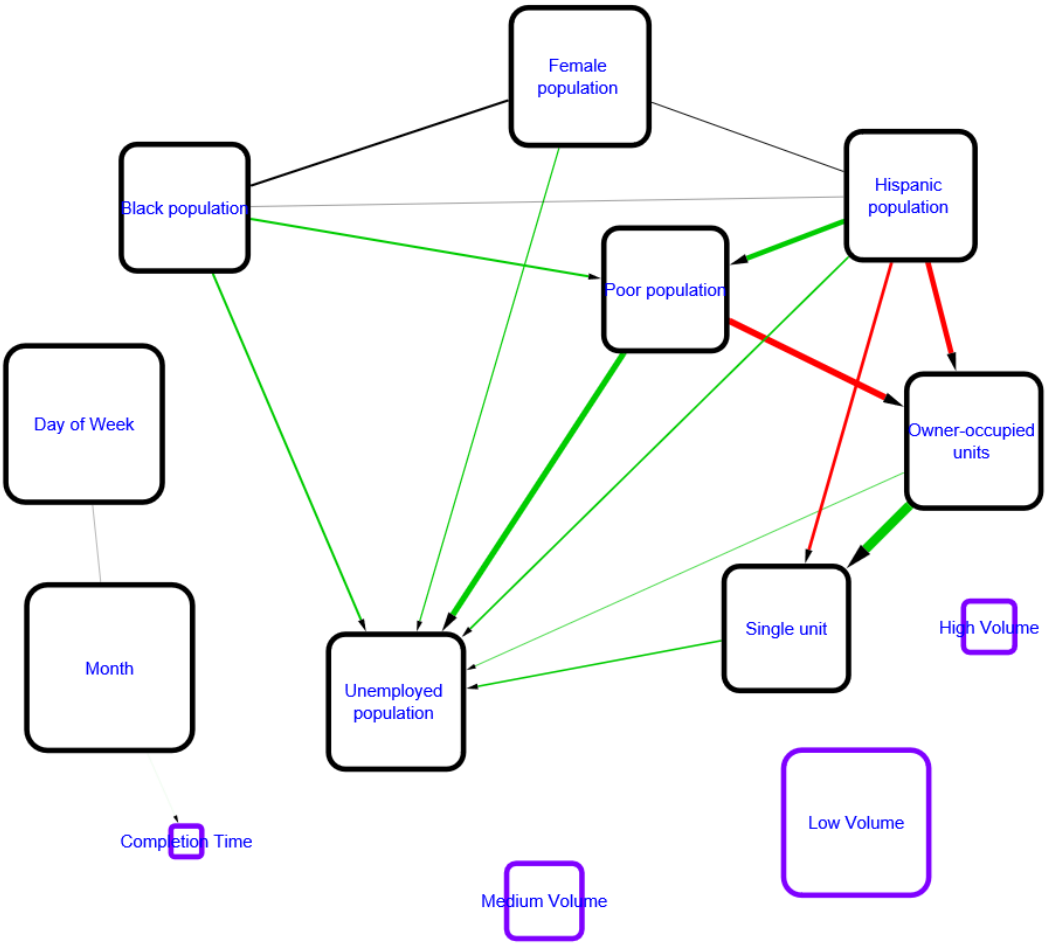


Figure 5.10: Signed Bayesian network generated from San Francisco 311 datasets for the years ranging from 2013 to 2019 using all requests. The variables are aggregated by the block group, day of the week, and month of the year. The network suggests one weak temporal impact (target variable, i.e., completion time affected by the temporal variable) with regard to government responsiveness, but with very low statistical significance.

impractical in many situations due to time, expense, and/or ethical concerns. It is therefore natural to consider ways to infer causal relationships from observational data. Fortunately, this is possible under certain assumptions using recent advances in *causal inferencing*. The networks generated from the observational data suggest a structure for the network of cause-effect relationships among the variables. These algorithms, however, often fail to provide the magnitude of the causal impacts. Therefore, we first looked at the overall score of the network to evaluate the fitness of the data and focus on the causal effect value of other variables on the task completion time.

5.5.1 Bayesian Gaussian equivalent score

The maximum likelihood score for a network, computed as a posterior probability derived from the priors can be used as a measure of goodness for the network. This is referred as the Bayesian Gaussian equivalent(BGe) score [Geiger and Heckerman, 1994]. The expression of BGe score is quite complex, and it will not be discussed here.

Table 5.3: Bayesian Gaussian equivalent scores of the generated sBNs for all the three cities, i.e., Miami-Dade County, New York City, and San Francisco. Benchmark is the generated network without integrating any piror knowledge (blacklisting) and temporal variables i.e., day of the week and month of the year the request was made. Improved networks incorporate the prior knowledege and temporal variables.

	Miami-Dade County	**San Francisco**	**New York City**
Benchmark	-10925	-325424	-231365
Improved	**-8655**	**-310442**	**-203610**

We provide the BGe scores of the improved networks and the benchmark from Chapter 4) for comparison. We observe that including the improved networks fits the data better as the value is higher than the benchmark (5.3). Therefore, we can conclude that integrating

prior knowledge and adjusting the confidence interval can help us to improve the quality of the network and the inference process.

5.5.2 Causal effect

Another issue in causal inference is the need to quantify the causal impact on one variable on another directly from the sBN. While the edges and the conditional probability information at each node can help quantify the causal effect on a neighboring node (along with the statistical significance of the effect), the causal effect of one variable on another variable that is not an immediate neighbor is unclear. The causal effect of variable V_x on V_y is defined as the difference in V_y when V_x is changed by a unit amount [Sazal et al., 2020b] without changing any other variable. If such conditions had been observed, the causal impact could be readily computed. If that is not the case, it is impractical to run randomized controlled trials due to time, expense, and/or ethical concerns. Fortunately, a data-driven approach, IDA, was developed by Kalisch et al. [2012], which uses observational data to estimate bounds on causal effects under certain assumptions.

To quantify the causal influence of a variable V_x on V_y, first the value of V_y is computed by forcing V_x to take the value $V_x = x$. Next, the value of V_y is computed when the values of V_x is forced to take either $V_x = x + 1$ or $V_x = x + \delta$, where δ is a small change to the value of V_x. If V_x and V_y are both random variables, imposing $V_x = x$ could change the distribution of V_y. The resulting distribution after modification is represented by $P[V_y|\texttt{do}(V_x = x)]$, as suggested by Pearl [2009]. Using the $\texttt{do}$ operator, the causal effect is defined as $E[V_y|do(V_x = x + 1)] - E[V_y|do(V_x = x)]$, where E stands for the expected value of multivariate Gaussian random variables.

We computed the causal effect of demographic and socioeconomic variables on all three cities' service completion time. From the causal effect plot (Fig. 5.11), it is possible

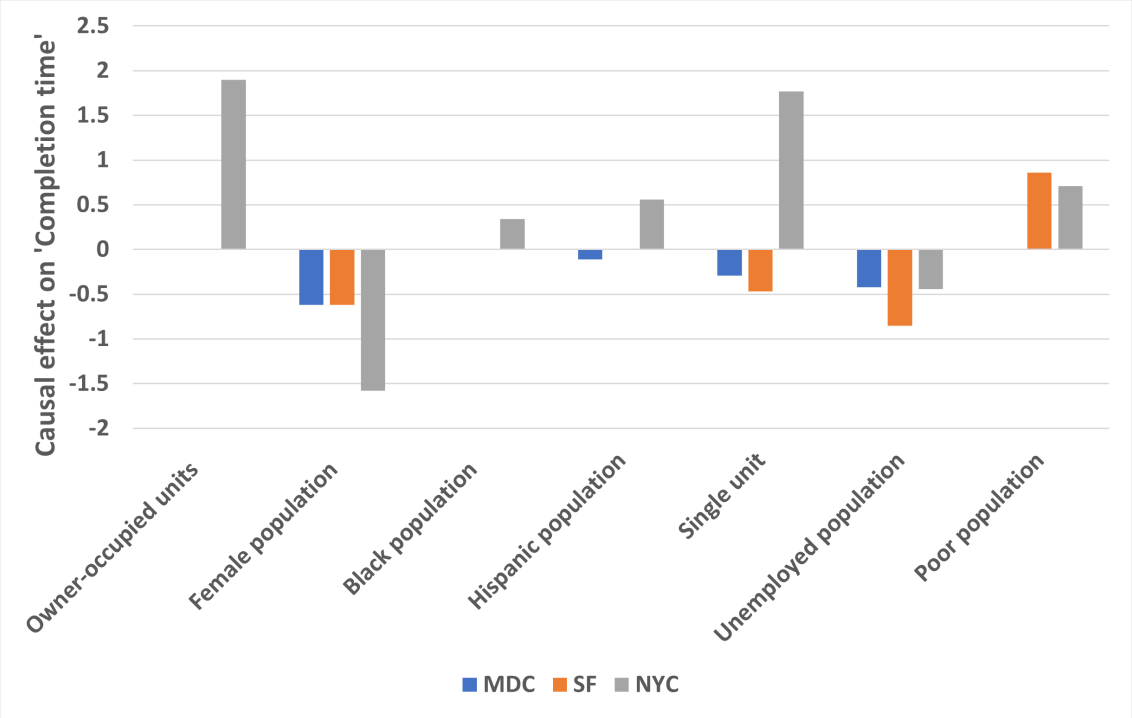


Figure 5.11: Comparison of causal effect of demographic and socioeconomic variables on "Completion time" for three different cities, i.e., Miami-Dade County, San Francisco, New York City.

to make several observations. The 'Owner-occupied units,' 'Black population,' 'Hispanic population,' 'Single units,' and 'Poor population' have a positive causal effect on the service completion time in NYC. The 'Female' and 'Unemployed population' negatively affect the service completion time in all the cities. 'Single units' in M-DC and SF have a negative causal effect on service completion time, unlike NYC. As a result, we observe some impact of demographic and socioeconomic variables on service completion time for all three cities from the causal effect analysis. However, the causal effect magnitudes are comparatively small in M-DC and SF compared to that in NYC.

The publicly available 311 dataset offers the opportunity for researchers to investigate the bias (if any) of government's responsiveness in providing non-emergency services to its citizens. The 311 centers provide an empirical case to assess the effectiveness of organizational innovations. We looked at how governments participate in or respond to requests, and in particular, how effectively they respond. We believe that looking into the government's response has ramifications beyond merely determining if one community receives better service than another. Knowing that the government is not adequately or fairly dealing with the neighborhood's problems weakens public trust in government institutions. As an example, emergency response failures following Hurricane Katrina were widely criticized for favoring the wealthy and white inhabitants over the poor and minority residents [Elliott and Pais, 2006]. We intended to learn more about whether these engagements, when mediated by technical interfaces like 311, result in equitable outcomes, with the goal of improving public trust in the institutions that serve them. This chapter looked at the demographic and socioeconomic factors that might influence the 311 centers' efficiency and equity in providing services. The main contribution was to examine the 311 data to extract meaningful information, depict it for policy decision-making, and propose a data analytics methodology and interpret descriptive results. We also reviewed the limitations of the 311 data analytics: Processes, Potential Benefits, and Limitations

of causal inference with the Bayesian network approach. We looked at data from three different cities, i.e., Miami-Dade County, New York City, and San Francisco, to study how effective the system was with regard to completion time. The causal (signed) Bayesian network models generated showed little or no impact of demographic and socioeconomic factors on the target variables, including completion time and call request volumes. We concluded that the findings do not support the existence of demographic or socioeconomic bias in providing non-emergency services to residents of these cities.

Table 5.4: Common columns for the three cities, i.e., Miami-Dade County, New York City, San Francisco 311 data.

Miami-Dade	**San Francisco**	**New York**	Desciption
ticket_id	CaseID	Unique Key	Unique identifier of a Service Request (SR) .
case_owner	Responsible Agency	Agency Name	Name of the responding agency.
case_owner_description		Agency	Acronym of responding agency.
issue_description	Request Details	Descriptor	Provides further details of the SR.
issue_type	Request Type	Complaint Type	Fist level of a hierarchy identifying the topic of the SR.
method_received	Source	Open Data Channel Type	Medium used to submit the SR. i.e., Phone, Online, Mobile, etc.
ticket_created_date_time	Opened	Created Date	Date SR was created.
ticket_last_update_date_time	Updated		SR closing date by agency.
ticket_closed_date_time	Closed	Closed Date	SR closing date by agency.
ticket_year	Opened	Created Date	SR created year.
ticket_status	Status	Status	Status of SR.
created_year_month	Opened	Created Date	SR created month
goal_days		Due Date	Expected date to update the SR.
sr_xcoordinate	Point	X Coordinate (State Plane)	X coordinate of the location (geo validated).
sr_ycoordinate	Point	Y Coordinate (State Plane)	Y coordinate of the location (geo validated).
latitude	Latitude	Latitude	Latitude of the location (geo validated).
longitude	Longitude	Longitude	Longitude of the location (geo validated).
city	Neighborhood	City	City of the location (geo validated).
street_address	Address	Incident Address	House number of incident.
Zip_code		Incident Zip	Incident location zip code (geo validated).
	Street	Street Name	Street name of incident location.
location_city		Location	lat & long of the incident location (geo validated).

Table 5.5: Additional columns for the three cities, i.e., Miami-Dade County, New York City, San Francisco 311 data.

City	Column	Description
Miami-Dade	state	Name of the state.
	actual_completed_days	Time taken to complete the SR.
San Francisco	Media URL	A URL to media associated with the request, i.e., an image.
	Neighborhood	SF Neighborhood.
	Supervisor District	SF Supervisor District.
	Status Notes	Explanation of status change or more details on current status.
	Category	The human understandable name of the SR type.
New York City	Resolution Action Updated Date	Date when responding agency last updated the SR.
	Resolution Description	Describes the last action taken on the SR by the responding agency.
	Cross Street 1	First cross street based incident location (geo validated).
	Cross Street 2	Second cross street incident location (geo validated).
	Intersection Street 1	First intersecting street incident location (geo validated).
	Intersection Street 2	Second intersecting street incident location (geo validated).
	Address Type	Type of incident location information available.
	Landmark	Name of the landmark (If incident location is as a Landmark).
	Facility Type	Type of city facility associated to the SR.
	Community Board	Provided by geovalidation.
	BBL	Borough Block and Lot, provided by geovalidation.
	Borough	Provided by the submitter and confirmed by geovalidation.
	Park Facility Name	Name of the facility (if the location is a Parks Dept facility).
	Park Borough	The borough of incident (if the location is a Parks Dept facility).
	Vehicle Type	Type of vehicle (If the incident is identified as a taxi).
	Taxi Company Borough	Borough of the taxi company (if in taxi).
	Taxi Pick Up Location	Pick up location (if in taxi).
	Bridge Highway Name	Name of the Bridge (if in Bridge/Highway).
	Bridge Highway Direction	Direction where the issue took place (if in Bridge/Highway).
	Road Ramp	Road or the Ramp (if in Bridge/Highway).
	Bridge Highway Segment	Additional information on the section (if in Bridge/Highway).

CHAPTER 6

CONCLUSION

As regulators, government agencies, and the general public become more reliant on AI-based dynamic systems, better accountability for decision-making processes will be essential to promote confidence and transparency in the field of public policy and administration. Many AI-based "black box" algorithms fail to be widely adopted due to their lack of interpretability. In an effort to contribute to the field of interpretability of AI, this dissertation focused on the adaptation of interpretable AI methods to problems from two different fields, i.e., storage systems and public policy.

Cache Replacement Modeled as a Variant of the Multi-armed Bandit (MAB) problem with Delayed Feedback and Decaying Costs: In Chapter 2, we formulated a new MAB variant with delayed feedback and decaying cost (MAB-DFDC) applicable to the cache replacement problem. In this variant, we assume that feedback can be delayed, the cost decreases with increasing delay in feedback, and the regret vanishes over time if the expected delay crosses a certain threshold. As a solution, we proposed the EXP4-DFDC algorithm and prove that expected regret is upper bounded by O($2\sqrt{2KT\ln N}$) for any learning rate η, where K is the number of possible actions, T is the number of rounds, and N is the number of experts. The regret bound guarantees a vanishing regret per round as T grows without bounds. Finally, we showed that the machine learning-based cache replacement algorithm LeCaR can be viewed as a simplified version of EXP4-DFDC. If the static learning rate of $\eta = 0.45$ is replaced with the derived theoretical optimal learning rate, $\eta_{OPT} = \min(1, \sqrt{\frac{K\ln N}{2}})$, the associated regret will be upper bounded by $R_{OLeCaR}(T) \leq 2\sqrt{2KT\ln N}$ [Yusuf et al., 2020, 2021b].

ALeCaR– adaptive version of LeCaR with learning rates adapting to the input: In the first application we studied, we set the stage for theoretical analysis of reinforcement learning applied to the cache replacement problem. The multi-armed bandit analogy will help researchers find optimal hyperparameters for regret minimization and choose an appropriate model. Even small improvements in cache optimization may lead to a significant boost in storage systems performance. For storage systems researchers, consistently designing high-performing caching algorithms remains an intriguing, but elusive, goal. ALeCaR achieves this goal by introducing a new family of machine-learned caching algorithms that are both lightweight and adaptive. ALeCaR is efficient because it allows the use of exactly two, potentially complementary, experts. For a variety of combinations of workload and cache size, ALeCaR with the suggested experts, LIRS and LFU, is the most consistently performing method. Furthermore, ALeCaR made it simple to combine a cutting-edge caching technique like ARC or DLIRS with a complementary expert like LFU to better manage a broader range of workloads.

Causal Inferencing and its Challenges: The Case of 311 Data: The 311 administrative dataset offers the opportunity for researchers to investigate the non-emergency services provided by local governments to local residents. We analyzed the data from Miami-Dade County to see how effective the system was in terms of completion time and call volume. We concluded that the data showed no evidence of demographic and/or socioeconomic bias in providing non-emergency services to Miami-Dade County residents. The case study using only one city's data cannot guarantee that data from other cities or municipalities will yield the same results. The findings, however, are in line with the previous study based on 311 data from other cities. Finally, we highlighted the difficulties of using the causal method to analyze this sort of dataset due to missing, impure, and inadequate data [Yusuf et al., 2021a].

Comparative analysis of 311 data for different cities: Following our discussion on the challenges of applying the causal (signed) Bayesian network approach to the 311 administrative datasets, we attempted to overcome those challenges in this task. First, we added temporal variables to the network to address the issue of confounding variables to a limited extent. Second, we added prior domain knowledge to blacklist edges going from the demographic and socioeconomic variables to the completion time and request volume variables. Next, we extended this framework to two more cities (New York City and San Francisco), where the number of observations is 2 to 10 times larger than the previous study (Miami-Dade County). We concluded that the data revealed no evidence of demographic or socioeconomic bias in providing non-emergency services to residents of all the three cities that we studied. More significantly, this research introduced a framework for applying the causal inference method on 311 datasets that may easily be expanded to data from other cities or similar types of administrative datasets, i.e., the N11 dataset.

6.1 Future work

6.1.1 Computer Systems

We envision putting ML into the hardware and improving hardware caches, thus bringing ML-based algorithms to manage CPU caches, which can bring enormous speedups to all computing systems. Other interesting problems include applying the methods developed here to variant cache models, i.e., non-datapath caches, which work with the assumption that caching every item accessed by a program can be counterproductive. Non-datapath caches allow for the option of not caching some of the items, thus saving on cache space and on cache writes and potentially improving speedups. When there is a cache miss, instead of choosing from one of the two policies (i.e., LRU and LFU), we envision designing

a variant of ALeCaR that will select from one of three choices: LIRS, LFU, or "avoid caching the item altogether".

Hierarchical Caches are organized in multiple levels. As we move up the hierarchy away from the processor, the cache levels become larger in capacity, slower in terms of access times, and less expensive. The resulting algorithm will be faced with many design choices, including deciding (a) the policy to be applied at each level, (b) the parameter choices for each level, and (c) choices of update functions for the parameters at each level. We can design a 2-level cache, and assume that the new algorithm will apply ALeCaR at each level with independent adaptation as the algorithm progresses. Hierarchical caches represent yet another variant on which our ML-based algorithms can be applied in the future.

6.1.2 Policy Analysis

Our approaches have far-reaching ramifications for government agencies, citizens, and researchers. For starters, the study will immediately assist customer service centers i.e., 211 and 311, to better understand the demand and needs of their citizens. The developed framework is not only applicable to 311 datasets, but also to the analysis of other related datasets, such as 211 and 911 datasets, and where causal inference can dramatically impact policy decisions. Also, part of our endeavors was to create a single emphData Hub for all the available 311 datatset. For this purpose, we collected data from many cities and standardized them. The data hub will assist not only in the research process, but also in distributing the study findings to a larger audience of professionals involved in the service delivery process, as well as to concerned citizens. Third, by curating the N11 "big data" repository, the project will help to enhance the sparse research on public sector organizations. The information will be used to kick-start fresh studies on public

and nonprofit sectors. Once the data hub is ready and fully functional, with our causal framework we will be able to expand the inference process on more cities with less manual intervention. We envision a local government that uses automated processes for all its decision-making and forecasting.

Given that adequacy of good data is an important issue and often a shortcoming, we can exploit the techniques of *Generative Adversarial Networks* (GAN) [Goodfellow et al., 2014] to generate data where very little data exists. In machine learning, generative modeling is an unsupervised learning approach that entails automatically identifying and learning regularities or patterns in input data such that the model may be used to create or output new instances that might have been drawn from the same dataset.

BIBLIOGRAPHY

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing ATARI with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Sam Ganzfried and Farzana Yusuf. Computing human-understandable strategies: deducing fundamental rules of poker strategy. *Games*, 8(4):49, 2017.

Min Chen, Yixue Hao, Kai Hwang, Lu Wang, and Lin Wang. Disease prediction by machine learning over big data from healthcare communities. *IEEE Access*, 5:8869–8879, 2017.

Andrew L Beam and Isaac S Kohane. Big data and machine learning in health care. *JAMA*, 319(13):1317–1318, 2018.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. LSTM neural networks for language modeling. In *Thirteenth Annual Aonference of the International Speech Communication Association*, 2012.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Judea Pearl. *Causality*. Cambridge University Press, 2009.

Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic books, 2018.

Randy Goebel, Ajay Chander, Katharina Holzinger, Freddy Lecue, Zeynep Akata, Simone Stumpf, Peter Kieseberg, and Andreas Holzinger. Explainable ai: the new 42? In *International cross-domain Conference for Machine Learning and Knowledge Extraction*, pages 295–303. Springer, 2018.

Andreas Holzinger. From machine learning to explainable ai. In *2018 World Symposium on Digital Intelligence for Systems and Machines (DISA)*, pages 55–66. IEEE, 2018.

Hani Hagras. Toward human-understandable, explainable AI. *Computer*, 51(9):28–36, 2018.

Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature, 2019.

Ismail Ari, Ahmed Amer, Robert B Gramacy, Ethan L Miller, Scott A Brandt, and Darrell DE Long. ACME: Adaptive caching using multiple experts. In *WDAS*, pages 143–158, 2002.

Pinchao Liu, Adnan Maruf, Farzana Beente Yusuf, Labiba Jahan, Hailu Xu, Boyuan Guan, Liting Hu, and Sitharama S Iyengar. Towards Adaptive Replication for Hot/Cold Blocks in HDFS using MemCached. In *2019 2nd International Conference on Data Intelligence and Security (ICDIS)*, pages 188–194. IEEE, 2019.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

George Amvrosiadis, Ali R. Butt, Vasily Tarasov, Erez Zadok, Ming Zhao, Irfan Ahmad, Remzi H. Arpaci-Dusseau, Feng Chen, Yiran Chen, Yong Chen, Yue Cheng, Vijay Chidambaram, Dilma Da Silva, Angela Demke-Brown, Peter Desnoyers, Jason Flinn, Xubin He, Song Jiang, Geoff Kuenning, Min Li, Carlos Maltzahn, Ethan L. Miller, Kathryn Mohror, Raju Rangaswami, Narasimha Reddy, David Rosenthal, Ali Saman Tosun, Nisha Talagala, Peter Varman, Sudharshan Vazhkudai, Avani Waldani, Xiaodong Zhang, Yiying Zhang, and Mai Zheng. Data Storage Research Vision 2025: Report on NSF Visioning Workshop. Technical report, USA, 2018.

Fernando J Corbato. A paging experiment with the multics system. Technical report, Massachusetts Inst Of Tech Cambridge Project, 1968.

S. Jiang and X. Zhang. LIRS: An efficient low inter-reference recency set replacement policy to improve buffer cache performance. In *Proc. ACM Sigmetrics Conf.*, pages 297–306, 2002.

N. Megiddo and D. S. Modha. ARC: A self-tuning, low overhead replacement cache. In *Proceedings of the 2nd USENIX Conference on File and Storage Technologies*, FAST '03, pages 115–130, Berkeley, CA, USA, 2003. USENIX Association. URL http://dl.acm.org/citation.cfm?id=1090694.1090708.

N. Megiddo and D. S. Modha. Outperforming LRU with an adaptive replacement cache algorithm. *IEEE Computer*, 37(4):58–65, 2004.

Giuseppe Vietri, Liana V. Rodriguez, Wendy A. Martinez, Steven Lyons, Jason Liu, Raju Rangaswami, Ming Zhao, and Giri Narasimhan. Driving Cache Replacement with ML-based LeCaR. In *Proc. of the USENIX Workshop on Hot Topics in Storage Systems (HotStorage)*, June 2018.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

Liana V. Rodriguez, Farzana Yusuf, Steven Lyons, Eysler Paz, Raju Rangaswami, Jason Liu, Ming Zhao, and Giri Narasimhan. Learning cache replacement with CACHEUS. In *19th USENIX Conference on File and Storage Technologies (FAST 21)*, pages 341–354. USENIX Association, February 2021. ISBN 978-1-939133-20-5. URL https://www.usenix.org/conference/fast21/presentation/rodriguez.

Cong Li. DLIRS: Improving low inter-reference recency set cache replacement policy with dynamics. In *Proceedings of the 11th ACM International Systems and Storage Conference*, SYSTOR '18, pages 59–64, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5849-1. doi: 10.1145/3211890.3211891. URL http://doi.acm.org/10.1145/3211890.3211891.

Herbert Robbins. Some aspects of the sequential design of experiments. In *Herbert Robbins Selected Papers*, pages 169–177. Springer, 1985.

Marcelo J Weinberger and Erik Ordentlich. On delayed prediction of individual sequences. *IEEE Transactions on Information Theory*, 48(7):1959–1976, 2002.

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 873–881, 2011.

John Langford, Alexander J Smola, and Martin Zinkevich. Slow learners are fast. *Advances in Neural Information Processing Systems*, 22:2331–2339, 2009.

Gergely Neu, Andras Antos, András György, and Csaba Szepesvári. Online markov decision processes under bandit feedback. In *Advances in Neural Information Processing Systems*, pages 1804–1812, 2010.

Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization. *The Journal of Machine Learning Research*, 15(1):3873–3923, 2014.

Miroslav Dudik, Daniel Hsu, Satyen Kale, Nikos Karampatziakis, John Langford, Lev Reyzin, and Tong Zhang. Efficient optimal learning for contextual bandits. *arXiv preprint arXiv:1106.2369*, 2011.

Herbert Robbins. Some aspects of the sequential design of experiments. *In Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

Nick Littlestone and Manfred K Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

Vladimir G Vovk. A game of prediction with expert advice. In *Proceedings of the Eighth Annual Conference on Computational Learning Theory*, pages 51–60. ACM, 1995.

Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of online learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37. Springer, 1995.

Yoav Freund and Robert E Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29(1-2):79–103, 1999.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 01 2002. doi: 10.1137/s0097539701398375.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 322–331. IEEE, 1995.

Chris Mesterharm. On-line learning with delayed label feedback. In *International Conference on Algorithmic Learning Theory*, pages 399–413. Springer, 2005.

Jon Christian Mesterharm. *Improving on-line learning*. Rutgers The State University of New Jersey-New Brunswick, 2007.

Alina Beygelzimer, John Langford, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandit algorithms with supervised learning guarantees. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 19–26, 2011.

Pooria Joulani, Andras Gyorgy, and Csaba Szepesvári. Online learning under delayed feedback. In *International Conference on Machine Learning*, pages 1453–1461, 2013.

Ofer Dekel, Elad Hazan, and Tomer Koren. The blinded bandit: Learning with adaptive feedback. In *Advances in Neural Information Processing Systems*, pages 1610–1618, 2014.

Baruch Awerbuch and Robert Kleinberg. Competitive collaborative learning. *Journal of Computer and System Sciences*, 74(8):1271–1288, 2008.

Balazs Szorenyi, Róbert Busa-Fekete, István Hegedus, Róbert Ormándi, Márk Jelasity, and Balázs Kégl. Gossip-based distributed stochastic bandit algorithms. In *International Conference on Machine Learning*, pages 19–27. PMLR, 2013.

Nicolo Cesa-Bianchi, Claudio Gentile, Yishay Mansour, and Alberto Minora. Delay and cooperation in nonstochastic bandits. *Journal of Machine Learning Research*, 49: 605–622, 2016.

Pooria Joulani, András György, and Csaba Szepesvári. Delay-Tolerant online convex optimization: Unified analysis and Adaptive-Gradient algorithms. In *AAAI*, volume 16, pages 1744–1750, 2016.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006. URL `https://books.google.com/books?hl=en&lr=&id=zDnRBlazhfYC&oi=fnd&pg=PA1&dq=+Prediction_Learning_and_Games&ots=6ARtfRBPsJ&sig=VUsDBbKhZ6J5SpItXOmY1P8l60w`.

Donghee Lee, Jongmoo Choi, Jong-Hun Kim, Sam H Noh, Sang Lyul Min, Yookun Cho, and Chong Sang Kim. LRFU: A spectrum of policies that subsumes the least recently used and least frequently used policies. *IEEE transactions on Computers*, (12): 1352–1361, 2001.

Walter L Smith. Regenerative stochastic processes. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 232(1188):6–31, 1955.

A Khachaturyan, S Semenovskaya, and B Vainstein. Statistical-thermodynamic approach to determination of structure amplitude phases. *Sov. Phys. Crystallography*, 24(5):519–524, 1979.

L-W Chan and Frank Fallside. An adaptive training algorithm for back propagation networks. *Computer speech & language*, 2(3-4):205–218, 1987.

Roberto Battiti. Accelerated backpropagation learning: Two optimization methods. *Complex Systems*, 3(4):331–342, 1989.

Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. Routledge, 2017.

Stuart J Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education Limited, 2016.

Sam Ganzfried and Farzana Yusuf. Optimal weighting for exam composition. *Education Sciences*, 8(1):36, 2018.

Sam Ganzfried and Farzana Beente Yusuf. Optimal number of choices in rating contexts. *Big Data and Cognitive Computing*, 3(3):48, 2019.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.

James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, pages 2546–2554, 2011.

VP Plagianakos, GD Magoulas, and MN Vrahatis. Learning rate adaptation in stochastic gradient descent. In *Advances in Convex Analysis and Global Optimization*, pages 433–444. Springer, 2001.

Jang-Hee Yoo, Jae-Woo Kim, and Jong-Uk Choi. An adaptive training method of back-propagation algorithm.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul): 2121–2159, 2011.

Matthew D Zeiler. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

SNIA. SNIA block I/O traces. `http://iotta.snia.org/tracetypes/3`.

D Arteaga and M. Zhao. VISA lab block I/O traces. `http://visa.lab.asu.edu/web/resources/traces`.

Dushyanth Narayanan, Austin Donnelly, and Antony Rowstron. Write off-loading: Practical power management for enterprise storage. *ACM Transactions on Storage (TOS)*, 4(3):10, 2008.

Deng Zhou, Wen Pan, Wei Wang, and Tao Xie. I/O characteristics of smartphone applications and their implications for eMMC design. In *2015 IEEE International Symposium on Workload Characterization*, pages 12–21. IEEE, 2015.

Akshat Verma, Ricardo Koller, Luis Useche, and Raju Rangaswami. SRCMap: Energy proportional storage using dynamic consolidation. In *FAST*, volume 10, pages 267–280, 2010.

Dulcardo Arteaga and Ming Zhao. Client-side flash caching for cloud systems. In *Proceedings of International Conference on Systems and Storage*, SYSTOR, pages 7:1–7:11, New York, NY, USA, 2014. ACM.

Ricardo Santana, Steven Lyons, Ricardo Koller, Raju Rangaswami, and Jason Liu. To arc or not to arc. In *HotStorage*, 2015.

Gil Einziger, Ohad Eytan, Roy Friedman, and Ben Manes. Adaptive software cache management. In *Proceedings of the 19th International Middleware Conference*, pages 94–106. ACM, 2018.

Jerry L Hintze and Ray D Nelson. Violin plots: A box plot-density trace synergism. *The American Statistician*, 52(2):181–184, 1998.

Jacob Cohen. *Statistical power analysis for the behavioral sciences*. Academic press, 2013.

Daniel Navarro. *Learning statistics with R*. Lulu. com, 2015.

John Clayton Thomas. Citizen, customer, partner: Rethinking the place of the public in public management. *Public Administration Review*, 73(6):786–796, 2013.

Mark Steyvers, Joshua B Tenenbaum, Eric-Jan Wagenmakers, and Ben Blum. Inferring causal networks from observations and interventions. *Cognitive Science*, 27(3):453–489, 2003.

Nir Friedman, Michal Linial, Iftach Nachman, and Dana Pe'er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7(3-4):601–620, 2000.

Xiao-Fei Zhang, Le Ou-Yang, and Hong Yan. Incorporating prior information into differential network analysis using non-paranormal graphical models. *Bioinformatics*, 33 (16):2436–2445, 2017.

Musfiqur Rahman Sazal, Daniel Ruiz-Perez, Trevor Cickovski, and Giri Narasimhan. Inferring relationships in microbiomes from signed Bayesian networks. In *2018 IEEE 8th International Conference on Computational Advances in Bio and Medical Sciences (ICCABS)*, pages 1–1. IEEE, 2018.

Musfiqur Sazal, Kalai Mathee, Daniel Ruiz-Perez, Trevor Cickovski, and Giri Narasimhan. Inferring directional relationships in microbial communities using signed Bayesian networks. *BMC Genomics*, 21(6):1–11, 2020a.

Musfiqur Rahman Sazal, Vitalii Stebliankin, Kalai Mathee, and Giri Narasimhan. Causal inference in microbiomes using intervention calculus. *BioRxiv*, 2020b.

Dimitris Margaritis. Learning Bayesian network model structure from data. Technical report, Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science, 2003.

Silvia Acid, Luis M de Campos, Juan M Fernández-Luna, Susana Rodrıguez, José Marıa Rodrıguez, and José Luis Salcedo. A comparison of learning algorithms for Bayesian networks: a case study based on data from an emergency medical service. *Artificial Intelligence in Medicine*, 30(3):215–232, 2004.

Chang-Ju Lee and Kun Jai Lee. Application of Bayesian network to the probabilistic risk assessment of nuclear waste disposal. *Reliability Engineering & System Safety*, 91(5): 515–532, 2006.

Baoping Cai, Yonghong Liu, Zengkai Liu, Xiaojie Tian, Yanzhen Zhang, and Renjie Ji. Application of Bayesian networks in quantitative risk assessment of subsea blowout preventer operations. *Risk Analysis*, 33(7):1293–1311, 2013.

Yahya Y Bayraktarli, Jens-Peder Ulfkjaer, Ufuk Yazgan, and Michael H Faber. On the application of Bayesian probabilistic networks for earthquake risk management. In *9th International Conference on Structural Safety and Reliability (ICOSSAR 05)*, pages 20–23, 2005.

Karen Sachs, David Gifford, Tommi Jaakkola, Peter Sorger, and Douglas A Lauffenburger. Bayesian network approach to cell signaling pathway modeling. *Science's STKE*, 2002 (148):38, 2002.

Pilar Fuster-Parra, P Tauler, M Bennasar-Veny, A Ligkeza, AA Lopez-Gonzalez, and A Aguilo. Bayesian network modeling: A case study of an epidemiologic system analysis of cardiovascular risk. *Computer Methods and Programs in Biomedicine*, 126:128–142, 2016.

Nicholas J Harding. *Application of Bayesian networks to problems within obesity epidemiology*. PhD thesis, The University of Manchester (United Kingdom), 2011.

Arnold Ojugo and Oghenevwede Debby Otakore. Forging an optimized Bayesian network model with selected parameters for detection of the coronavirus in delta state of nigeria. *Journal of Applied Science, Engineering, Technology, and Education*, 3(1):37–45, 2020.

Anna Rigosi, Paul Hanson, David P Hamilton, Matthew Hipsey, James A Rusak, Julie Bois, Karin Sparber, Ingrid Chorus, Andrew J Watkinson, Boqiang Qin, et al. Determining the probability of cyanobacterial blooms: the application of bayesian networks in multiple lake systems. *Ecological Applications*, 25(1):186–199, 2015.

Linwood D Hudson, Bryan S Ware, Kathryn B Laskey, and Suzanne M Mahoney. An application of bayesian networks to antiterrorism risk management for military planners. Technical report, George Mason University, 2005.

Michail Tsagris. A new scalable Bayesian network learning algorithm with applications to economics. *Computational Economics*, pages 1–27, 2020.

Chee Kian Leong. Credit risk scoring with Bayesian network models. *Computational Economics*, 47(3):423–446, 2016.

Eric W Welch, Charles C Hinnant, and M Jae Moon. Linking citizen satisfaction with e-government and trust in government. *Journal of Public Administration Research and Theory*, 15(3):371–391, 2005.

Benjamin Y Clark, Jeffrey L Brudney, and Sung-Gheel Jang. Coproduction of government services and the new information technology: Investigating the distributional biases. *Public Administration Review*, 73(5):687–701, 2013.

Benjamin Y Clark et al. Do 311 systems shape citizen satisfaction with local government? *SSRN 2491034*, 2014.

Benjamin Y Clark, Jeffrey L Brudney, Sung-Gheel Jang, and Bradford Davy. Do Advanced Information Technologies Produce Equitable Government Responses in Co-production: An Examination of 311 Systems in 15 US Cities. *The American Review of Public Administration*, 50(3):315–327, 2020.

James R Elliott and Jeremy Pais. Race, class, and hurricane katrina: Social differences in human responses to disaster. *Social Science Research*, 35(2):295–321, 2006.

Lingjing Wang, Cheng Qian, Philipp Kats, Constantine Kontokosta, and Stanislav Sobolevsky. Structure of 311 service requests as a signature of urban location. *PLoS One*, 12(10):e0186314, 2017.

Taewoo Nam and Theresa A Pardo. Identifying success factors and challenges of 311-driven service integration: a comparative case study of NYC311 and Philly311. In *Proceedings of the 46th Hawaii international conference on system sciences*, 2013.

Sarah Hartmann, Agnes Mainka, and Wolfgang G Stock. Citizen relationship management in local governments: The potential of 311 for public service delivery. In *Beyond Bureaucracy*, pages 337–353. Springer, 2017.

Yuchen Li, Ayaz Hyder, Lauren T Southerland, Gretchen Hammond, Adam Porr, and Harvey J Miller. 311 service requests as indicators of neighborhood distress and opioid use disorder. *Scientific reports*, 10(1):1–11, 2020.

Yilong Zha and Manuela Veloso. Profiling and prediction of non-emergency calls in New York City. In *Proceedings of the Workshop on Semantic Cities: Beyond Open Data to Models, Standards and Reasoning, AAAI*, 2014.

Constantine Kontokosta, Boyeong Hong, and Kristi Korsberg. Equity in 311 reporting: Understanding socio-spatial differentials in the propensity to complain. *arXiv preprint arXiv:1710.02452*, 2017.

Li Xu, Mei-Po Kwan, Sara McLafferty, and Shaowen Wang. Predicting demand for 311 non-emergency municipal services: An adaptive space-time kernel approach. *Applied geography*, 89:133–141, 2017.

Xian Gao. Learning within the 311 service policy community: Conceptual framework and case study of Kansas City 311 program. 2018.

Corey Kewei Xu and Tian Tang. Closing the gap or widening the divide: The impacts of Technology-Enabled coproduction on equity in public service delivery. *Public Administration Review*, 2020.

Nader Madkour. *Predicting Non-Emergency 311 Requests for an Efficient Resource Allocation After a Disaster in Houston, Tx*. PhD thesis, Lamar University-Beaumont, 2020.

Christopher W Zobel, Milad Baghersad, and Yang Zhang. Calling 311: evaluating the performance of municipal services after disasters. In *ISCRAM*, 2017.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT Press, 2009.

Judea Pearl, F Bacchus, P Spirtes, C Glymour, and R Scheines. Probabilistic reasoning in intelligent systems: Networks of plausible inference. 1995.

Kevin B Korb and Ann E Nicholson. *Bayesian artificial intelligence*. CRC Press, 2010.

Marco Scutari. Bayesian network constraint-based structure learning algorithms: Parallel and optimised implementations in the bnlearn R package. *arXiv preprint arXiv:1406.7648*, 2014.

Marco Scutari and Jean-Baptiste Denis. *Bayesian networks: with examples in R*. CRC press, 2014.

Thomas Verma and Judea Pearl. An algorithm for deciding if a set of observed independencies has a causal explanation. In *Uncertainty in Artificial Intelligence*, pages 323–330. Elsevier, 1992.

Markus Kalisch, Martin Mächler, Diego Colombo, Marloes H Maathuis, and Peter Bühlmann. Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software*, 47(11):1–26, 2012.

P Spirtes, C Glymour, and R Scheines. Causation, prediction and search (MIT Press, Cambridge). 1993.

Diego Colombo and Marloes H Maathuis. Order-independent constraint-based causal structure learning. *The Journal of Machine Learning Research*, 15(1):3741–3782, 2014.

Diego Colombo, Marloes H Maathuis, Markus Kalisch, and Thomas S Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, pages 294–321, 2012.

Miami-Dade County open data. Miami-dade county open data hub. `https://gis-mdc.opendata.arcgis.com`.

Def. by ACS. Poverty definition by census. `https://www.census.gov/quickfacts/fact/note/US/IPE120219`.

Marco Scutari. Learning Bayesian networks with the bnlearn R package. *arXiv preprint arXiv:0908.3817*, 2009.

Mitch Fernandez, Juan D Riveros, Michael Campos, Kalai Mathee, and Giri Narasimhan. Microbial "social networks". *BMC genomics*, 16(11):S6, 2015.

Nir Friedman, Moises Goldszmidt, and Abraham Wyner. Data analysis with Bayesian networks: A bootstrap approach. *arXiv preprint arXiv:1301.6695*, 2013.

Female-headed black family statistics in the US. Female-headed black family statistics. `https://www.statista.com/statistics/205106/number-of-black-families-with-a-female-householder-in-the-us`.

Farzana Yusuf, Shaoming Cheng, Sukumar Ganapati, and Giri Narasimhan. Causal Inference Methods and their Challenges: The Case of 311 Data. In *DG.O 2021: The 22nd Annual International Conference on Digital Government Research*, pages 49–59, 2021a.

New York City open data. New york city 311 data hub. `https://nycopendata.socrata.com/Social-Services/311-Service-Requests-from-2010-to-Present/erm2-nwe9`.

San Francisco 311 data. San francisco 311 data. https://data.sfgov.org/City-Infrastructure/311-Cases/vw6y-z8j6.

Dan Geiger and David Heckerman. Learning Gaussian networks. In *Uncertainty Proceedings 1994*, pages 235–243. Elsevier, 1994.

Farzana Beente Yusuf, Camilo Valdes, Vitalii Stebliankin, Giuseppe Vietri, and Giri Narasimhan. Exp4-dfdc: A non-stochastic multi-armed bandit for cache replacement. *arXiv preprint arXiv:2009.11330*, 2020.

Farzana Beente Yusuf, Vitalii Stebliankin, Giuseppe Vietri, and Giri Narasimhan. Cache replacement as a mab with delayed feedback and decaying costs, 2021b.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

VITA

FARZANA BEENTE YUSUF

2016–2021	Ph.D., Computer Science Florida International University Miami, Florida, USA
2016–2019	M.Sc., Computer Science Florida International University Miami, Florida, USA
2008–2013	B.Sc., Computer Science & Engineering Bangladesh University of Engineering & Technology ,Dhaka, Bangladesh
2018–2021	Graduate Research Assistant Algorithms for Machine Learning, Data Analytics, & Computing Systems Group Florida International University Miami, Florida, USA
2016–2017	Graduate Research Assistant Strategic Adversarial Multiagent Artificial Intelligence Lab Florida International University Miami, Florida, USA
2019	Summer Research Intern Search Ads Team Google Inc., USA
2015–2016	Team Lead Data Analytics Team Dohatec New Media, Dhaka, Bangladesh
2013–2015	Software Engineer Dohatec New Media Dhaka, Bangladesh
2016–2018	Graduate Teaching Assistant Florida International University Miami, Florida, USA

PUBLICATIONS

Farzana Yusuf, Shaoming Cheng, Sukumar Ganapati and Giri Narasimhan. “Causal Inference Methods and their Challenges: The Case of 311 Data”. In Proceedings of 22nd Annual International Conference on Digital Government Research (2021).

Liana Valdes*, Farzana Yusuf*, Steven Lyons, Eysler Paz, Raju Rangaswami, Jason Liu, Ming Zhao, Giri Narasimhan. “Learning Cache Replacement with CACHEUS”. In Proceedings of 19th USENIX conference on File and Storage Technologies, (2021), *The first two authors contributed equally to this work.

Farzana Yusuf, Camilo Valdes, Vitalii Stebliankin, Giuseppe Vietri, Giri Narasimhan. “EXP4-DFDC: A Non-stochastic Multi-Armed Bandit for Cache Replacement”. LXAI, NeurIPS workshop for LatinX in AI Research (2019).

Sam Ganzfried and Farzana Yusuf. “Optimal Number of Choices in Rating Contexts”. Big Data and Cognitive Computing (2019).

Pinchao Liu, Adnan Maruf, Farzana Yusuf, Labiba Jahan, Hailu Xu, Boyuan Guan, Liting Hu, Sitharama S. Iyengar. “Towards adaptive replication for Hot/Cold blocks in HDFS using Memcached”. In Proceedings of International Conference on Data Intelligence and Security (2019).

Sam Ganzfried and Farzana Yusuf. “Optimal Weighting for Exam Composition”. Education Sciences (2018).

Sam Ganzfried and Farzana Yusuf. “Computing Human-Understandable Strategies: Deducing Fundamental Rules of Poker Strategy”. Games (2017).

Sam Ganzfried and Farzana Yusuf. “Learning Human-Understandable Strategies”. AAAI Workshop on Computer Poker and Imperfect Information Games (2017).

Farzana Yusuf, Camilo Valdes, Vitalii Stebliankin, Giuseppe Vietri, Giri Narasimhan. “EXP4-DFDC: A Non-stochastic Multi-Armed Bandit for Cache Replacement”. Research day, SCIS, FIU.

Farzana Yusuf, Camilo Valdes, Vitalii Stebliankin, Giuseppe Vietri, Giri Narasimhan. “EXP4-DFDC: A Non-stochastic Multi-Armed Bandit for Cache Replacement”. Three Minutes Thesis, CEC, FIU.