Big Data Management and Mining Models and their Applications

by

Anifat Mojirola Olawoyin

A thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada

July 02, 2024

Thesis advisor
Dr. Carson K. Leung

Author
Anifat Mojirola Olawoyin

Big Data Management and Mining Models and their Applications

Abstract

The world is dynamic, so are big data. The evolving challenges of managing big data volume, variety, veracity, validity, and velocity has resulted in several studies focusing on solving one or more of these perplexing issues. In this Ph.D. research, I focus on the evolving issues arising from big data variety, veracity, privacy, and accessibility. First, I design a conceptual model for capturing and storing variety of big data types including structured, semi-structured and unstructured data types and in addition, design a metadata collection framework for managing the big data in support of machine learning and open data FAIR principle of Findable, Accessibility, Interoperability and Re-usability such that the information about the data are available beyond the life cycle of the data. Second, I design hierarchical spatial-temporal model (HSTM) for managing individual record in big data in the aforementioned open data lake architecture with metadata collection framework. Third, I extend the HSTM and design the resulting hierarchical spatial-temporal privacy preserving model (HSTPPM) for preserving privacy of individual record in big data. Fourth, I extend and design applications of the HSTPPM to big data co-occurrence pattern mining and big data visualization.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Publications

The following peer-reviewed publications are relevant to this thesis:

1. Preserving privacy of temporal big data [OLC20b]
2. Privacy preservation of COVID-19 contact tracing data [OLW21]
3. Privacy-preserving health-care analytics of trajectory data [LOW21]
4. Open data lake to support machine learning on Arctic big data [OLC21a]
5. Big data management for machine learning from big data [OLHC23]
6. Preserving Privacy integration and mining for big temporal co-occurrence patterns [OL22]
7. Privacy-preserving publishing and visualization of spatial-temporal information [OLC21b]
8. Privacy-preserving spatio-temporal patient data publishing [OLC20a]
9. Evolution of big data models from hierarchical models to knowledge graphs [OLC23]

Acknowledgements

My first appreciation goes to my academic supervisor—Dr. Carson K. Leung—for his tremendous advice, insightful encouragement, immeasurable discussion on various aspects of this thesis and several publications in the last four years. The journey with him started as my External Examiner for my M.Sc. degree at University of Winnipeg in 2019. He has overwhelmingly supported my growth and success as a student, a researcher, a sessional instructor, a public officer and a mother.

I would like to thank members of my Ph.D. thesis examination committee—Dr. Yang Wang (Computer Science, who is currently at Concordia University), Dr. Carl Ngai Man Ho (Electrical and Computer Engineering), and Dr. Christiana Ijeoma Ezeife (University of Windsor)—for providing insightful expert review and recommendations to further improve the thesis work after the candidacy exam. Thanks Dr. Xihui Larry Liang (Mechanical Engineering) for chairing my Ph.D. oral examination.

I also appreciate Connor Hryhoruk, Qi Wen and other members of the Database and Data Mining Lab at the University of Manitoba, especially those who have contributed in the dozens of conference and journal publications.

I am indebted to Dr. Alfredo Cuzzocrea (University of Calabria, Italy) for countless collaborations on conference and journal publications in the last four years.

Special appreciation to my academic supervisor, Mitacs, and Arctic Research Foundation (ARF) for their financial support on this work.

It has been my pleasure to work with Garry Casey, Christine Cox, Tom Henheffer, Donald McLennan, and Svein Vagle (ARF); Gagandeep Kaur Chahal, Muhammad Khan, Chong Li, Hongming Li, Patience Ngcobo, Chukwuma Ogu, Nicholas Logan Olson, Salweyar Patel, Tushar Sharma, and Jing Xiao led by Sharmain Donovan, Ralph Dueck, Reynard Dela Torre, and Gwen Zwaagstra (RRC Polytechnic); as well as Eddy Carmack (University of Victoria) on a joint university-college-industry project during my Ph.D. study.

Special thanks to professors at University of Winnipeg—Dr. Sergio Camonlinga, Dr. Yangjun Chen, Dr. Christopher Henry (who recently joined University of Manitoba), and Dr. Sheela Ramanna—for their support & encouragement towards the publication of my first research paper in 2018 and for their recommendation for my current Ph.D. program.

Lastly, my deepest appreciation to my family, my spouse, my kids (Aisha, Aminat and Ayman), my siblings and my mother for their understanding, and support.

Anifat Mojirola Olawoyin
B.Sc.(Hons.), University of Winnipeg, Canada, 2016
M.Sc., University of Winnipeg, Canada, 2019

The University of Manitoba
July 02, 2024

To my mother, Alirat Anike

Chapter 1

Introduction

In recent time, the landscape of big data collection has evolved due to the availability of wide varieties of data sources enhanced by technology advancement. Big data are generated continuously through countless real-life activities all over the world. For instance, in healthcare, huge volume of patients' data is generated through the electronic medical record system (EMR) [HWH+24]. In retails, millions of data are collected in store and online transactions [LAA+23]. in Banking, unimaginable data are collected through customer interactions with different banking technologies and platforms such as the automated teller machine (ATM), online banking, telephone banking and in recent time, more data are collected through the banking app enhanced by the convenience and trust of mobile phone system by majority of bank customers. Millions of data are collected by government agencies on various transactions such as employment insurance, childcare subsidy, wage subsidy, childcare benefits, application for permanent residence, citizenship application and most recently pandemic benefits. Hydrographic and bathymetric data are collected by Arctic researchers through expedition activities; Social network is another source of big data collection through countless human interactions including messages, chat, emails, blogs, and video conferencing. Thus, big data remains an active research area for interdisciplinary fields including machine learning [SW11], database system [LÖ18], data visualization [MFDW17, PVS+18], data mining [CASL17, LÖ18, SW11, HPT22], data analytics [Shi22, LBH+19], epidemiology and genomic [SL19], information re-

trieval, knowledge discovery [WFG+24] and bio-informatics [TKIH19]. The general characteristics of big data using the 7 V's are described in the following section.

1.1 General Characteristics of Big Data

In general, big data can be characterized by their well-known multiple V's. The common 7 V's are described below:

1. **Variety:** The type and format of the big data are many due to the nature of data collection resulting from ever growing internet of things (IoT), the environment and specialty requirements of various professional bodies such as the geologist, surveyors, financial analyst, Engineers, and others. Types and formats of the big data can be:

    - Structured, in which case data can be easily organized in rows and columns of a relational database table or excel spreadsheet.
    - Semi-structured, which can be represented using the extensible markup language (XML), Hypertext markup language (HTML) or JavaScript object notation (JSON).
    - Unstructured, which can be challenging to represent due to non-organization characteristics. Format for unstructured data includes Portable Document format (PDF), *.txt for text data, *.png, *.jpg for image data, Extended Triton Format (*.xtf). audio, video, graph, charts, and links to other web pages.

    Information retrieval, data mining and analytics are non-trivial for semi-structure and unstructured data because preprocessing such as data cleaning, feature selection and/or data transformation are required for the tasks. Integration of these variety of big data from heterogeneous sources remain an active research area. *Thus, the first challenge for the thesis is the design of a conceptual model for storing variety of big data from heterogeneous sources.* In addition, the thesis aims at investigating a data driven approach for capturing data about

the data known as metadata. Metadata supports effective management of big data by providing users with data about the data, enabling search of related data and providing another basis for scientific investigation, machine learning training and prediction on the metadata and promoting reusability of data.

2. **Veracity:** Big data may be precise or imprecise (Uncertain). The degree of uncertainty varies greatly depending on several factors including change in weather condition and inherent environmental hazards in some cases due to the terrain characteristics of the data collection areas such as the arctic. Precise data are consistently measured and are mostly accurately defined. Precise data include retail store transactions, banking, and insurance transactions. Uncertain data [JL15, CLZW16] are estimated data characterized by inconsistency in measurement with varying degree of accuracy such as hospital lab test report, blood pressure, social network, arctic expedition data and sensor data. In an attempt to capture hidden pattern in uncertain data, scientists have developed several algorithms and models which can be classified into three approaches namely:

   - Traditional methods such as query processing,
   - Data mining algorithms such as classification and clustering; and
   - Transform based techniques, example in this category includes various machine and deep learning models.

   This thesis will focus on designing an algorithm for managing specific area of uncertain big data known as spatio-temporal data. spatio-temporal data can be classified into four categories- event data, trajectories data, point reference data and raster data. In all classification, spatio-temporal data are rooted in space and time and are specifically different from other data due to the presence of spatial and temporal attributes. Time is continuous; objects are related in space; spatio-temporal data collection can thus be fixed when done at specific time in specific space or may be dynamic and continuous such as atmospheric temperature collected for weather forecast. Spatio-temporal data mining and analysis are particularly useful in real life such as crime pattern detection, near-

est neighbor measurement, community detection, trends and pattern of infectious diseases, environmental science among other uses.

3. **Volume:** Big data are collected in large quantities in numerous real-life applications such as transportation network [LBH+19], finance [SMÜ19], social networks [JLT12], biomedical data [SL19, Leu24], and trajectory data [ZXM+10] among others. The exponential growth in internet of things (IoT) usage, availability, and trust of low-cost mobile devices for big data collection are some of the driving forces for the ever-increasing volume of data generated and collected from heterogeneous sources. While the volume of data keeps increasing, the major challenge is how society, government and researchers can leverage the voluminous data without jeopardizing the privacy of individuals. Hence, another challenge for this thesis is the design of privacy preserving model to preserve privacy of individual in the huge volume of big data and at the same time compress the volume of data to a manageable level without jeopardizing the utility of the data.

4. **Velocity:** The world is dynamic, so are big data. Velocity of big data refer to rate of change of different aspect of data life cycle management including temporal change in data collection volume, change in data structure and database schema as well as change in use and reuse.

   First, data collection rate varies largely depending on purpose of collection and types of activities. For instance, the rate of data collection for social network data [LJ15, JLP16] will be different from the data collected for disease reports [TKIH19], genomics data [SL19], and epidemiological statistics [DKS+24]. Some professional bodies have standard for data collection rate, for example, sound velocity profile (SVP) data collected during arctic or ocean expedition is measured in meter per second, whereas the social media data are mainly driven by human interactions such as web search, messaging, texting, video and image sharing on social media platforms motivated by availability of low cost high quality smartphones and can be measured in megabyte per second, per minute

or per hour.

Secondly, big data velocity can be described in term of change in volume of data collection at different temporal level. For instance, business often defines peak and low sales hour, day, month or seasonal, then, there is also peak period associated with data collection activities having deadlines such as closing date for student admission application, rush hours for city transit due to people resuming for work between 7am and 9am and so on. Subsequently, due to the temporal variation in data collection volume, the transaction write operation or load also has peak and low rate which eventually affects processing time and cost.

Lastly, velocity of big data can be viewed in term of rate of change in data structure and schema restructuring due to the ever-changing business requirements. Data are often collected for a purpose; however, some un-needed data are collected in the process of collecting purposeful data and such data may become useful later especially when some hidden patterns are revealed in the process of data mining, data analytics or through other data analysis tasks. Along the same line of dynamism of use and purpose, some purposeful collected data may be discarded in the process. Subsequently, the rate of change in structure and schema may result in high operation and database maintenance cost.

5. **Visualization:** The flood of big data available in today's world has necessitated the need for visualization to provide actionable insights including outliers, flow of disease spread, human mobility, storm weather forecast and other environmental hazard monitoring to enhance proactive decision making and policy planning. In the past, most research focused on time and space efficiency to reduce processing time and space requirement for storing ever increasing volume of big data. However, the research focus has recently shifted to data analytics in form of visualization [Leu23, DGLT23] also known as visual analytics. Traditionally, data are stored in relational database and using query processing combined with statistics metrics to retrieve information for further analysis and processing. However, the task of mining hidden pattern from the growing

Table 1.1: COVID-19 contact tracing sample data

Date	City	Latitude	Longitude	Province	Frequency
2020-02-16	Andong-si	36.5625775	128.7180250	Gyeongsangbuk-do	5
2020-02-16	Cheonan-si	36.7694416	127.2363829	Chungcheongnam-do	4
2020-02-16	Daegu	35.8714354	128.6014450	-	21
2020-02-16	Wonju-si	37.3736804	127.9579728	Gangwon-do	6
2020-02-16	Incheon	37.4212689	126.6475892	-	5

volume of data has generated lot of concerns including the rigid requirement of relational database of having a well-structured schema design before storing the data. Subsequently, data warehouse evolves over time expanding the schema concept of relational database to star schema of storing large volume of data in several tables and having a central table to capture the relationships among several tables. However, data warehouse lack agility because it requires fixed schema and does not provide visualization tools. The possibility of having data storage and data visualization tools that can capture dynamic and complex relationship among interacting entities in a densely connected social network has resulted in graph database research and subsequently what is known today as knowledge graph. For instance, let us consider a subset of South Korea COVID-19 contact tracing data pattern for February 26, 2020 with more than three frequency count in the same city, province and having the same (x, y)-coordinates to determine cluster of infection for the specific date. As shown in Figure 1.1, the cluster of infection is better represented in the graph visualization while the relational database query output in Table 1.1 provided the detail information:

```
SELECT date, city, type, latitude, longitude, province,
       count(*) as frequency
FROM patientroute
WHERE date='2020-02-16'
GROUP BY date, city, type, latitude, longitude, province
HAVING count(*) > 3
```

6. **Validity:** While the volume of big data is growing exponentially, the adoption of machine and deep learning to process the data as fast as possible and help

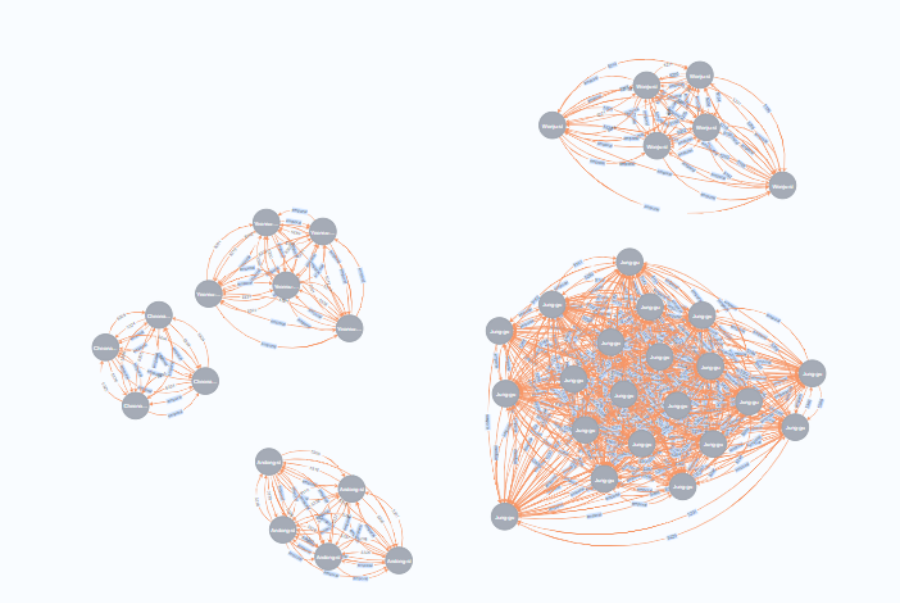


Figure 1.1: SK COVID-19 sample infection clusters

make decision on the data is also growing in various sectors including credit ratings and loan processing, hiring decision, re-offense risk assessment in policing, voice, and image recognition. Meanwhile, the machine and deep learning are black box models, the output and decision from these models are sometimes questionable, unreliable, biased, and difficult to understand by human. For instance, a criminal risk assessment machine learning prediction model designed by the Correctional Management Profiling for Alternative Sanctions (COMPAS) was proven to be racial biased in decision outcome by Angwin et al. [ALMK16] using statistical measure of false positive rate while Kilbertus et al. [KRP+17] and Chiappa et al. [CI19] argued that using statistical fairness as a measure of discrimination may not be appropriate and proposed causal reasoning approach. Despite the proliferation in the argument of metrics to sufficiently establish the validity of output from machine models, the facts remain that machine and deep learning models' output are questionable and often require more explanation for human understanding. Hence, there are lot of ongoing research on interpretability and fairness of machine and deep learning models [MST20, LK21, MKG+20, LFM+21].

7. **Value:** Big data has huge value in numerous complex real world scenarios specifically providing opportunities to use past and present data to model future prediction, resource allocation as well as proactive prevention and mitigation of societal issues such as crime prevention, future pandemic planning and preventing spread of infectious diseases using big data collection for contact tracing, restricting movement of people from known epicenter of disease outbreak, demographic pattern analysis of outbreak to aid production of drugs and vaccine for specific demographic age and gender, detecting community spread of diseases among other use. Big data are also useful for academic researchers for scientific investigation, learning and discovery. In addition, big data are a relevant tool for policy makers for proactive planning for the future, improved governance, and citizen engagement. Meanwhile, the general public benefits from the enhanced big data literacy through open data initiatives.

1.2 Challenges of Big Data Management

There are challenges in managing big data, which include those listed in the following subsections.

1.2.1 Complex Varieties

An obvious challenge of big data is complexity of capturing, storing, and modeling evolving varieties of big data. The advancement in technology has resulted in collection of semi-structured and unstructured big data in different domains including telecommunication, social network [LJ15] and biological data [YKK17] that are difficult to model using the predominant relational database. For instance, in relational database structured traditional data can be captured using tables having rows representing data records and each column is an attribute. Relationships among entities are defined using referencing such as one-to-one, one-to-many or many-to-many and the tables can be queried using structured query language (SQL). One-to-one and one-to-many relationships are simple to represent using primary key as a foreign

key in the other table; however, a more complex relationship such as many-to-many requires creation of a relation table to model the relationship. Thus, tabular representation poses a challenge as the relationship among various entities of the big data increases. The relationship representation becomes difficult to manage and may require schema restructuring. As a result, query processing may become time consuming and expensive.

Over time, other data storage and models have evolved to resolve diversity of big data type variety issue such as the document database, NoSQL database and recently, the need to adequately model and visualize dynamic and complex relationship among entities of a network-based data such as telecommunication, biochemical interaction and social network has triggered a novel area of research known as graph database. Graph database combines concept of graph theories [BC12] from the field of mathematics with database, data mining [LMJ14, RAL19, LH13] and information retrieval to model complex relationship among various entities in the social network, telecommunication network, customer purchase pattern, information security network design, biochemical interaction of genes, human evolution, churn prediction [KSK20], neighborhood detection, crime, and other societal issues.

1.2.2 Uncertain Veracity

Another challenge of big data is the variation in the degree of uncertainty inherent in most data collections. For instance, data such as hydrographic and bathymetric data collected during arctic expedition, patient trajectories collected for COVID-19 contact tracing, sensor data, social network data, traffic flow data, smartphone data and continuous weather data are spatio-temporal in nature rooted in continuous space and time. Storing spatio-temporal data in relational database is not an issue, the challenge is spatio-temporal data mining, the non-trivial task of extracting previously unknown pattern to prevent, solve or predict future societal issues such as spread of COVID-19 and other infectious disease, drug development and proactive patient testing, storm prediction, crime detection in Police, signal loss and intrusion detection in communication and cyber attack monitoring. The traditional data mining methods

are sometimes restricted to sampling methods with the assumption that samples are independent and stationary [OC18], whereas, spatio-temporal are highly structural correlated in space and time. For instance, ocean and arctic expedition data are collected using a *moving* research vessel supported with a rigid-hulled inflatable boat (RHIB), sound velocity profiler, multi-beam echo sounder and antennas. Additionally, the weather condition varies from one expedition to another and hence, the data collection may not be replicable. Thus, domain knowledge and expertise are needed to preprocess, design and transform the raw spatio-temporal data before applying the data mining techniques. For instance, the Society of Exploration Geophysicists (SEG) developed an open standard SEG-Y file format, for storing geophysical data; Triton imaging incorporation developed an industrial standard known as Extended Triton Format (XTF) for recording side scan and back-scatter data collected during arctic expedition and the list of such standards keeps growing by the day. Hence, uncertainty data mining [JL15, LMT14] research has emerged and remains an active research area.

1.2.3 Privacy and Accessibility

Privacy and accessibility can be considered as double-edged sword for big data research. Nowadays, accessibility of data through social network as well as government and non-government promotion of open data concept in support of transparency in governance has created yet another issue in the era of big data. The social notion of privacy is defined by Hornung et al. [HMS+17] as a dynamic social process of negotiating personal boundaries and territories, managing disclosure, concealment and relationship depending on individuals, contexts and situations. However, *the boundary of personal privacy in the public domain is currently undefined.*

Westin [Wes68] defined privacy as the right of individuals, institutions or group to control when, how and to what extent information about them is communicated to others. In other word, preserving privacy is a necessity in the entire life cycle of big data management starting from data collection, data storage, data processing, data analysis, data publishing to the point where data are eventually destroyed. Chal-

lenges of data acquisition includes issues such as term of use, government legislation, copyright, disclaimers, and privacy. For instance, Statistics Canada [1] data collection and reuse is guided by government legislation such as the *Access to Information Act* and the *Privacy Act* while special permits are sometimes required to meet legal requirements in cases where the data are sensitive and proprietary. In addition, some publicly available data include copyright rules for data reuse, content ownership and disclaimers are often included on social media platform to isolate platform content from individual contributor's content for instance, X (formerly Twitter)'s term of service [2] exclusively states:

> *"You are responsible for your use of the Services and for any Content you provide, including compliance with applicable laws, rules, and regulations."*

The term public consent has emerged as a justification for accessible open data and reusability and yet there is growing concern over what constitutes privacy of individuals within the publicly available data [OLC21b]. Though, open data publishing does involve anonymization of personal information such as social insurance number, names, and date of birth among others that could directly identify an individual, research has shown that despite removal of sensitive attributes, re-identification is not impossible. For instance, Sweeney [Swe02] demonstrated how individual could be identified from the United States 1990 census data with *US 5-digits Zip code* using infrequent pattern modeling technique and in particular, Sweeney identified the governor of Massachusetts by linking common attributes as shown in Figure 1.2 from anonymous voters' list and hospital level data collected by Massachusetts Group Insurance Commission (GIC) containing attributes such as gender, birth date, ethnicity and Zip code. Due to the possibility of re-identification of individual record, several studies have investigated and designed anonymization models including k-anonymity [Swe02], l-diversity [MKGV07] and t-closeness[LLV07]. However, a major drawback of these models is the inherent utility-privacy trade-off such that strong privacy often resulted in minimal utility. Subsequently, another field of research known as differential privacy [ABCP13, XX15, CYXX17] emerged recommending data distortion

[1] https://www.statcan.gc.ca/
[2] https://twitter.com/en/tos

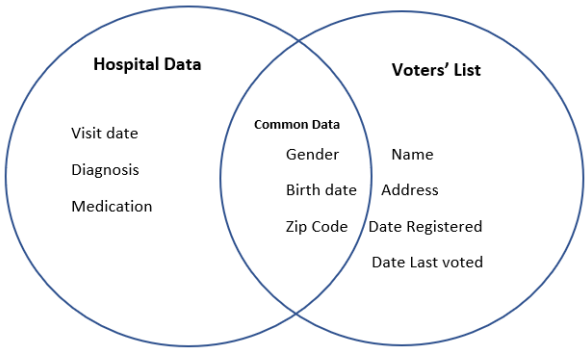


Figure 1.2: Data linking

through noise addition. However, adding noise to data may distort the data beyond its usefulness [OLC21b]. Thus, preserving privacy of individual record within big data remains an active research area.

1.2.4 Questionable Validity

Objectivity and accuracy are other issues for big data. Boyd and Crawford [BC12] defined big data as a scholarly phenomenon resting on the interaction of technology, analysis and mythology of truth, objectivity, and accuracy. However, we are far from the objectivity and accuracy of big data due to interpretability and fairness risks associated with machine and deep learning models use in processing big data. In an attempt to solve big data processing time issues arising from the ever increasing volume of big data that are infeasible to be processed by human effort, several machine and deep learning models have been developed including artificial neural networks (ARN) [OC18, LMW14, LEM$^+$20], deep neural networks (DNN) [LAL$^+$21, SML$^+$21], convolutional neural network (CNN) [LLY$^+$21] and most recently graph neural network (GNN) [ZBM$^+$22]. The validity of decision outcome of machine models is sometimes questionable; as an example, a quarantine message was sent in error to over ten thousand travelers by the Canada boarder agency non-GPS AI ArriveCan app recently. The goal of using the app is to collect travelers' information in advance of arriving in Canada to verify travelers' compliance with the public health emergency order and determine if a passenger must quarantine on arrival, hence, the decision to

quarantine is made at a point in time in the app data collection life cycle management using the information provided by the travelers in combination with certain decision support model. In addition, these models are often subjective to the inherent pattern in the dataset or underlying nature of statistics measure. For instance, while some researchers [LMKA16] argued that the four statistics metrics- true positive (TP), true negative (TN), false positive (FP) and false negative (FN) can be combined to measure fairness and bias of machine learning models, other researchers [MKG+20, CI19] have suggested causal reasoning approach and some are of the opinion that counterfactual [MST20, LK21] analysis approach is more appropriate. Meanwhile, models are created and trained by human using past and present data to predict the future pattern, hence, irrespective of the methods of explaining the output of machine models, big data collection are not error free. Duplication, inconsistency, missing words and labels, natural disaster, change in weather condition among other errors are inherent in the heterogeneous sources of data collection. Further to the complexity of bias in machine output and inherent inaccuracy of data sources, human involvement in the process of data cleaning, feature selection and other preprocessing tasks is subjective in nature and may introduce bias to the input even before processing by the machine model. Moreover, interpretation of machine learning output by different domain remains a subjective endeavor relying on the purpose of use. Thus, bias, fairness, accuracy, and subjectivity are issues that will most likely linger into the future.

1.3 Motivation and Objectives of the Thesis Work

Motivated by the prevailing challenges of big data despite the overarching usefulness, this thesis will focus on big data variety, veracity, privacy, and accessibility issues. With regards to variety, several models and concepts have evolved over time aiming at capturing varieties of big data in different format including relational database models for storing, entities, entities relationships and defined constraints for ensuring integrity of relationship among entities and reducing redundancy of data as much as possible. Subsequently, document database, extensible markup language

(XML), NoSQL databases emerged for storing and modeling semi-structured data. Yet, the growing variety of data in the world today are not static, hence this thesis aim to design a conceptual model for capturing and storing different variety of big data including structured, semi-structured and unstructured data along with metadata describing the data in support of open data FAIR principle of Findable, Accessibility, Interoperability and Re-usability. The conceptual model is schema-less repository for variety of raw data. The thesis aim to design a meta data collection framework for managing the big data stored in the data repository.

The second issue to be addressed by this thesis is privacy of big data by designing a privacy preserving model for spatio-temporal uncertain data (veracity) by extending the geographical notion of nearness to temporal relationship. How are objects, events, multi-values trajectories, point reference or raster spatio-temporal data related in space and time. The relationship between the different spatio-temporal data can be of three types: spatial relationship, temporal relationship, or spatio-temporal relationship. For instance, the trajectories of patient data collected for the purpose of contact tracing could be related to other data in space and time if released publicly. Such other spatio-temporal are many in our society including:

- Retail: Customers visiting grocery stores located at a particular city between specific time periods.
- Transportation: Transit pick, drop and stop location and time
- Airline: Travelers check in and check out from one airport to another all over the world.
- healthcare: Time and location of patient admission to hospitals, lab test, childbirth among other health spatio-temporal data.
- By-law enforcement: Parking tickets, speed limit violation, red light photo enforcement and so on.

For example, let us consider the subset dataset in Tables 1.2 and 1.3. Note that Table 1.2 shows an hospital level data about patient visit while Table 1.3 shows some parking

Table 1.2: Hospital visit data

Date	Hospital	Location	Gender	Diagnosis
3/12/2018 9:30 pm	Victoria	Winnipeg	Male	Flu
3/12/2018 3:30 pm	St Boniface	Winnipeg	Female	Fever
3/12/2018 12:30 am	Health Science	Winnipeg	Male	Diabetes
3/12/2018 8:45 am	Seven Oaks	Winnipeg	Female	Flu
3/12/2018 10:20 am	Seven Oaks	Winnipeg	Male	Diabetes
3/12/2018 12:10 pm	Victoria	Winnipeg	Female	HIV
3/12/2018 6:15 pm	Health Science	Winnipeg	Male	Bronchitis
3/12/2018 7:30 pm	Health Science	Winnipeg	Male	Diabetes

Table 1.3: Parking data

Parking date	Plate number	Geolocation
3/12/2018 11:58:20 AM	LPV1452	(49.903375°, -97.160025°)
3/12/2018 03:52:23 AM	PNN2589	(49.879116°, -97.150051°)
3/12/2018 01:58:18 PM	ADB2543	(49.898365°, -97.144276°)
3/12/2018 03:36:25 AM	TRR8547	(49.897101°, -97.148546°)
3/12/2018 03:38:02 AM	GFS4587	(49.879000°, -97.150190°)
3/12/2018 12:34:11 PM	WDS7893	(49.897043°, -97.140581°)
3/12/2018 03:53:05 AM	QLJ3605	(49.885185°, -97.143220°)

transactions. The temporal data in both datasets could be linked by an adversary with background knowledge of, for instance, the plate number of an individual and plotting the coordinate on a map will reveal the location which could be linked to the location information in the hospital data. Though the knowledge gained from linking the two tables is based on statistical or probability inference that the person diagnosed with a disease is the same person known to the adversary as the owner of the plate number in Table 1.3; aggregating the data would prevent such inference.

Individual trajectory of daily activities can as well be linked together, As an example, consider activities of an individual for a particular day, which:

- starts with a visit to a pharmaceutical store,
- a retail store,
- a health insurance agency,

- then parking; and
- ends the day with airline booking.

Though, these activities are disjointed, they can be linked using temporal and/or spatial attributes from each activity. In most cases, individual records may not be relevant in data analysis. A pharmacist and a store manager, for example, are more interested in the shopper purchase patterns probably for restocking of inventory. Similarly, travel agencies are more interested in booking patterns for promotional purposes of some tourist attractions and/or tourist activities. Then, the question is how we can extract useful hidden pattern from these uncertain data without jeopardizing privacy of individual record among other questions. The challenge for this thesis is to design a privacy preserving model to preserve both the spatial and temporal elements of spatio-temporal data such that the knowledge of an adversary is limited to the underlying aggregated pattern within the data itself, the location and temporal information that could be linked with other data on the web are privacy preserved. The model is anticipated to be a useful preprocessing step for many other big data management tasks such as data publishing, data mining, machine and deep learning prediction models and visual analytics. Aside from preserving privacy, the algorithm design aims to also help in data compression (volume) in support of space efficiency for big data storage. In addition, integrating different data from heterogeneous sources becomes trivial with the proposed model.

1.4 Thesis Statement

Motivated by the prevailing challenges of big data management discussed in this chapter, my Ph.D. thesis statement (which is also the major goal for this thesis) is to design big data management and mining models and apply these models to real-life scenarios. To elaborate, first, this thesis aims at designing a suitable repository that can accept ever growing variety of data including structured, semi-structured and unstructured data in their raw form without jeopardizing integrity of the data. Secondly, several privacy preserving models have been developed in the past focusing

on what is generally considered as personal identifiers such as names, date of birth and social insurance numbers and other quasi-identifiers which may indirectly related to a person such as zip codes in the United State and postal codes in Canada. However, these models failed to identify that linkage, exposure or de-identification is possible using spatio-temporal data because we are related in time and space. Thus, *key contributions* of this thesis include:

- Design of an open data lake architecture, which is a conceptual model for data repository that is anticipated to accept any form of data types as a solution to data variety issue and in addition design a methodology for collecting data about data known as metadata to effectively managed these big data acquired from various life activities including health care, social networks, government transactions, camera surveillance and bylaw enforcement among others.

- Design of a hierarchical spatio-temporal model (HSTM), which supports managing and utilizing spatio-temporal data in the open data lake architecture with metadata collection framework.

- Design of a hierarchical privacy preserving spatio-temporal model (HSTPPM), which extends the HSTM to support preserving spatio-temporal data while keeping a good balance of utility and privacy.

- Design of a data analytics solution for mining co-occurrence patterns and a visual analytics solution for visualizing data and patterns. This can be considered as an extension of the HSTPPM to support real-life applications to various issues around the world today including COVID-19 data contact tracing, airline and human trajectory, city by-law enforcement on parking violations.

Some questions arising from the big data challenges earlier discussed in this chapter are:

Q1. How to integrate open data initiatives with data lakes?

Q2. How to manage open spatio-temporal data at different granularity levels? How

to link heterogeneous data from different rich data sources and at different granularity levels?

Q3. How to preserve privacy of spatio-temporal data? How to do so for privacy-preserving publishing? How to do so far COVID-19 contact tracing?

Q4. How to mine interesting patterns—such as co-occurrence patterns—from open spatio-temporal data at different granularity levels? How to visualize them?

1.5 Thesis Organization

The first chapter of this thesis is the introduction discussing the seven Vs of big data, challenges of big data and motivation for the study. The challenge of big data management has been published as a refereed book chapter for the International Conference on Advanced Information Networks and Application (AINA) 2023 [OLHC23].

The remainder of this thesis is organized as follows. Chapter 2 gives background and related works. In particular, the chapter reviews and compares big data types, database models (e.g., data lake), open data principle, and privacy-preserving models. Part of this work has been published as a refereed paper in the 47th IEEE Annual Computer, Software and Applications Conference (COMPSAC) 2023 [OLC23].

Chapter 3 describes the design and implementation of an open data lake architecture and metadata collection framework. The evaluation and implementation of the design was done using real-world open data for Arctic research. This work has been published as a refereed paper in IEEE International Conference on Big Data 2021 [OLC21a].

Chapter 4 explores how to manage spatial data and temporal data in the open data lake architecture and metadata collection framework. The chapter describes the design and implementation of hierarchical spatio-temporal model (HSTM), which can be considered as a non-trivial integration of spatial and temporal hierarchical models for managing spatial and/or temporal data at different granularity levels. The chapter also explores different spatial representative points for representing spatial locations of temporal sequences of data points.

Chapter 5 extends the HSTM to preserve privacy of spatial and/or temporal data at different granularity levels in the HSTM. The chapter describes the design and implementation of the hierarchical spatio-temporal privacy-preserving model (HSTPPM) including its application to privacy-preserving publishing [OLC20a] and contact tracing [OLW21]. Moreover, the chapter also explores different privacy preserving enhancements—such as differential privacy (DP) and generative adversarial network (GAN)—to the HSTPPM and result in DP-HSTPPM and GAN-HSTPPM. Part of this work has been published as a refereed paper in the IEEE International Conference on Big Data 2020 [OLC20b].

Chapter 6 adapts the HSTPPM for data analytics and visual analytics tasks. Specifically, the chapter describes the adaptation of the HSTPPM for mining co-occurrence patterns and visualizing data/resulting patterns. This work has been published as refereed papers in the IEEE International Conference on Big Data 2021 and 2022 [OLC21b, OL22].

Finally, Chapter 7 draws conclusions and discusses future research directions.

Chapter 2

Background and Related Works

This chapter presents the discussion of big data management models relevant to this thesis. One of the motivations for this thesis is big data variety issues, hence, the chapter first examines the three general data types namely structured, semi-structured and unstructured. Next, the chapter presents various database and models that have evolved over time aiming at solving big data variety issues, then, compare and contrast some of the models on the basis of how the inherent challenges in one model resulted in the evolution of another model and in particular, how the society dynamic nature has affected the most notable and successful relational database models.

2.1 Big Data Types Variety

This section presents the three general classification for big data *variety*. First, *structured data type* refers to data that can be represented in tabular or flat format. Such data are represented in rigid format using columns and rows, data management tasks are easy and trivial nonetheless of astronomical growth in big data collection. Modeling structured data may be as simple as using excel, pivot chart, pivot tables and more advanced statistical models like regression or auto-regressive integrated moving average (ARIMA). However, storing structured data in relational database requires structural definition for the attributes, entity types and their relationships in addition to defining constraints governing the existence of the data within the

database such as the key constraint to uniquely identity each row of data, the entity constraint defining the primary key of an entity and the referential integrity constraints to maintain consistency of data across relationship participating entities.

Second, *semi-structured data* type refers to the resulting data type from the advent of the world wide web such that the data may not necessarily have a definite identical structure or format. Semi-structured data collection may be transient and dynamic and as a result defining relational structure for such data are non-trivial. Semi-structure data motivated the development of key value pair structure in which the data attributes, types, relationship, and value co-exist. Attributes of data object may be unknown in advance and can be added on the go. Semi-structured data are represented using graph or tree-like structure such as the extensible markup language (XML) data model.

Third, *unstructured data type* refers to data that are characterized mostly by uncertainty and irregularity. Data such as video, audio, image, portable document format (PDF), text messages, tweet, web pages and pictures with no specific pattern. The question arising from the growth of unstructured data collection over years is that if the data are unstructured why do we care about building a structure for it? The obvious answer is that there are hidden previously unknown pattern within those unstructured data that are useful for decision making. For instance, a typical pattern in crime investigation by the police since the advent of home-use surveillance camera is to ask neighbors to allow police access to the surveillance data. The before and after crime spatio-temporal data from the surveillance camera are particularly useful in identifying person of interest for further investigation.

2.2 Database and Database Models

This section presents the historical evolution of databases and database models relevant to big data management. Database is known to be a self-describing storage system while the database model describes the theories and concepts underlying the structure of the database and set of possible operations on the database. The typical structure of a database has three main components namely (a) Data types

describing the format of data values domain such as string, integer and boolean, (b) Relationship describing the interaction between and among entities within the database and (c) Constraints which are the restrictions that must hold at all times in the database for two obvious reasons; first, constraints are defined to maintain the integrity of data and secondly, constraints are essential security tool to prevent unauthorized use and misuse of data in the database. Two basic operations are essential for any database—updates and retrieval. Updates operation can be further divided into insertion, deletion, and modification.

In the 1960s, prior to relational database, the International Business Machine (IBM) corporation has introduced the concept of hierarchical modeling as part of the database management system designed for the commercial information management system (IMS). Tsichritzis and Lochovsky [TL76] described the facilities provided by IMS including the concept of hierarchical modeling of physical storage of data. Bachman [Bac69] described tree and network models as special types of graphical data structure techniques; the tree is a parent-child structure with only one root node, hierarchical model is a typical types of tree data structure. In contrast, the network structure permits unlimited interactions among nodes. Taylor and Frank [TF76] described the network model based on physical level and file system. The 1960s models prior to relational database model are described in Sections 2.2.1 and 2.2.2.

2.2.1 Network Model

The formal description of the Network model emerged from conference of data system languages (CODASYL) group. The model has a record type representing a data and a set type representing one-to-many relationship relating a record to many record instances using a pointer. A low-level network data manipulating language (DML) was proposed in 1971 by database task group (DBRG) as an extension of COBOL language. DML has several commands for adding, manipulating, and traversing records including (a) The STORE command used to insert new record type, (b) The $CONNECTTO$ command for associating a record type to a one-to-many set type, (c) Data retrieval commands such as FIND ANY, FIND DUPLICATE

and (d) Set type traversing commands like GET FIRST, GET NEXT and GET LAST WITHIN MEMBER for traversing data.

2.2.2 Hierarchical Model

The proponents of hierarchical model viewed the real-world applications and scenarios as naturally hierarchical in nature—from country government structure of presidential, monarchy or parliamentary, companies' management structure, military and judicial governance systems, university's governance structure, complex biological system such as ecological pyramid, ecclesiastical religion governance and even the family tree structure are built on hierarchical foundation. Thus, hierarchical structure is simple to implement using ***tree*** structure that can depict *parent, children, ancestor and descendant* kind of relationship as shown in Figure 2.1. For instance, if we consider a subset of university governance shown in Figure 2.2, *the university president is the ancestor, vice president is the parent, deans of faculties are the children while faculty and staff are example of descendants.* In essence, hierarchical model defines set of entities, a tree-like relationship connecting the entities using a one-to-one or one-to-many and since a child must have a parent, there is only one relationship between any two given entities (E_1, E_2). Information retrieval in hierarchical model is usually through tree traversal. For instance, Knuth [Knu14] defined a pre-order tree traversal as record traversal from top to bottom and left to right order such that retrieving *department chair* information from Figure 2.2 will include hierarchical path: *University President* $\rightarrow$ *Vice President Academy* $\rightarrow$ *Dean of Faculties* $\rightarrow$ *Department Chair*. The low-level record-at-a-time data manipulation language (DML) for the hierarchical model known as $DL/1$ has commands to navigate the tree structure such as GET FIRST, GET NEXT, GET NEXT WITHIN PARENT as well as commands to insert and replace record.

2.2.3 Shortcoming of the Hierarchical and Network Models

The shortcoming of the hierarchical and network models includes:

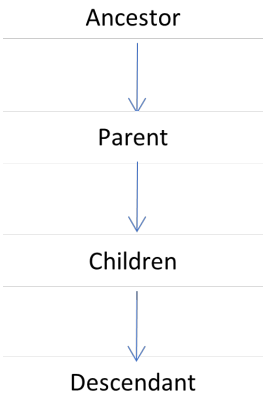


Figure 2.1: Hierarchical model

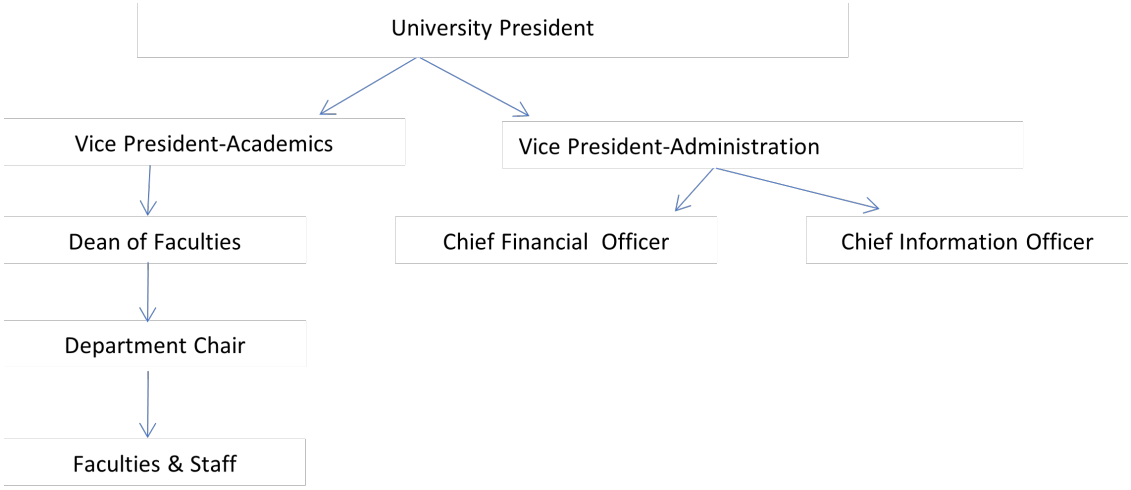


Figure 2.2: University hierarchical model

- Non-separation of conceptual representation and physical storage of data resulting in time-consuming and expensive maintenance particularly when new requirements are added requiring reorganization of the database and designing of more complex queries.

- Implementing many-to-many relationship is a non-trivial task involving duplication of two or more one-to-many relationship and hence two separate trees are required. For instance, if we consider real world scenarios where a company may register in many provinces and a province may register many companies;

then province and company are related with many-to-many *register* relationship; We can either have province as the parent (root) of the hierarchical tree and duplicate company name under each province of registration. Alternatively, we may have two separate trees, one showing one-to-many relationship for *Province* → *Company* in which case a province can register many companies and another tree showing one-to-many relationship for *Company* → *Province* where a company can register in many provinces.

- Database operation such as record insertion and deletion may become difficult to implement due to the rigid hierarchical order of the tree structure such that deletion of the root record for instance, require deletion of all the descendants.

- Additionally, the network and the hierarchical database models were accessed only through programming interfaces and as a result writing, testing and debugging new programs are common to implement new queries to answer users reporting requests, indicating that the models do not provide complete logical and physical data independence as done by the relational model to be discussed in subsection 2.2.4

2.2.4 Relational Database Model

Codd [Cod70] introduced the concept of relational database in 1970 based on set theory and relation separating the physical storage and conceptual representation of data. Set is essentially collection of objects and relation is a set of n-tuples or rows representing set of atomic values from a domain. The underlying motivation for the development of relational database was basically to provide a common baseline for data representation and efficient information retrieval using mathematics set theory. The relational model defines two constraints, first, the entity constraint defining the non-null primary key of an entity and ensuring uniqueness of tuple within the database and secondly, the referential integrity constraints to maintain consistency of data across multiple relations. Rather than using application interface like previous hierarchical and network models where the low-level DML must be embedded within

a general-purpose host programming language, high-level set-at-a-time Structured query language (SQL) was introduced as a common language for efficient reusable queries design and flexible database reorganization due to change or additional application requirements. The SQL language has standard statement for insertion (INSERT), updating (UPDATE), deleting (DELETE) and using the select (SELECT) statement in combination with basic logical operators of *AND, OR, NOT* and by extension the set operation of *union, intersection and difference* to retrieve and extract pattern of association among set of related entities. Relational model has evolved to become the leading model for most database system in the world today leveraging indexing and query optimization techniques for efficient information retrieval. Several commercial relational database managements exist today including Microsoft SQL, Oracle, INGRES and Sybase. However, schema design and modeling complex data structure remain major issues and active research areas in database system.

2.2.5 Semantic Database Model

Emerged in late 1970s and 1980s and motivated by lack of expressive conceptual data modeling from earlier database models that allow distinctive definition of data types, relationship and constraints that can be mapped to different databases. The argument for the semantic model school of thought relied on the fact that there are other database requirements that cannot be mapped to tabular representation of relational database and high-level conceptual model can be used for modeling such requirements without any reference to specific database implementation [PM88]. In other word, the semantic models aimed at designing high-level human perception of the real-world application requirements. one of the early semantic models is the entity-relationship (ER) conceptual model proposed by Chen [Che76, Che02] in 1976 based on two concepts (a) Entity representing real world objects, places or things and (b) Relationship describing interaction among entities, then entity has attributes which are the characteristics of an entity such name, date of birth, social insurance number and others. Chen's ER model is based on different mathematics theories including relation theory, logic, algebra, set theory and Lattice. The major differ-

ence between relational model and ER model can be viewed as explicit and implicit representation. While ER model used mathematics relation to express the structure of entities defining relationship set as mathematics relation on entity set; relational model used mathematics relation to express the structure of data values.

Chen's entity relationship (ER) model was further extended to include ternary relationship, composite attributes and norm forms [Lin85]; generalization and specialization abstraction, and introducing the concept of super-class, subclass and attribute inheritance. [TYF86, MS92, CERE90, SS77], aggregation [SS77], and entity-category-relationship (ECR) [NEL86]. A category is described as a subset of *union* between two or more distinct entities. Whereas the concept of generalization and specialization can be viewed as an hierarchical model having generalized entity at the root and specialization at the leaf of the tree hierarchy. For instance, an Employee may have one or more specialized skills or belong to a professional body such as Engineer, Accountant, Surveyor or Architect. Navathe's model recognized these groups in form of employee specialization. For instance, an *engineer* is a subclass and inherits all the attributes of employee (parent entity). Employee is a super class because engineer is an instance of entity type employee.

2.2.6 Object-Oriented Database Model

Object-oriented (OO) concept for database modeling evolved over time in the 1980s from three different research areas (a) programming language, (b) artificial intelligence, and (c) databases [Kim90]. Object oriented models represent data as collection of objects organized into classes with the notion of information hiding, type hierarchies, inheritance, polymorphism, or operator overloading and encapsulation. The intent for object-oriented database could be traced back to the object-oriented programming language designed to solve complex system requirements for engineering applications including computer graphics and computer-aided-design (CAD) engineering design and manufacturing software. Kim [Kim90] described the must-have features of an object-oriented database to include schema design, persistent storage and data definition language (DDL) for defining and modifying schema in addition to

creating and accessing objects in the database. From object-oriented programming perspective, OO database is designed to store persistent objects permanently enabling reusability and sharing of such objects by multiple programming language including Smalltalk, Java, Visual Basic, Delphi, Python, C++, and business application. Despite the benefits of the object-oriented approach in modeling complex data structure, allowing reusable persistent object sharing among programs and applications using concurrency control, and efficient information retrieval using indexing; object-oriented database still requires design of rigid schema like the relational database

2.2.7 Semi-structured Database Models

Buneman et al. [Bun97, BFW00] described semi-structured database model as graph-like or tree-like model motivated by (a) the emerging data from the World wide Web (WWW) that cannot be constrained in the relational database world because of the inherent irregularity and non-conformity to the existing common relational database schema, (b) non-flexibility of the existing relational and object-oriented query languages such the structural query language (SQL) to provide clauses for retrieving complex constraints on tree like path, and (c) integration of data from heterogeneous web sites was an aching issue in the 1990s demanding the attention of researchers in different areas of study including biologists looking for solution to capture molecular interactions and the growth of telecommunication demanding a model for capturing hierarchical layers of interconnected networks and also the evolution of the social network activated the need to think outside the rigid box of relational database to align with the reality that entities interactions and relationships are flexible, dynamic and not rigid. In addition, extensible markup language (XML) was introduced in the late 1990s as a standard for web and document data exchange and interoperability with hypertext markup language (HTML) [BPS+97]. XML combined the hierarchical structure and self-describing notion to enable parsing of the world wide web data. However, modeling complex relationship remains a challenge.

2.2.8 Graph Database Model

Visual representation, modeling highly connected social and telecommunication network data are some of the motivating factors for the emergence of graph database models. There exist few definitions of graph database in research literature; for instance, Angles and Gutierrez [AG08] defined graph database as a storage system for representing objects and inter-object relationships and including supports for efficient queries of object attributes such as finding shortest path or neighborhood using indexed lookup. Hence, efficient query processing, an important concept in relational database is also important in designing efficient graph database to enable evolving knowledge graph construction from the large data stored within the graph database. Macko et al. [MMS13] described graph database as a storage system in which the data structure and instances are modeled as graph and data manipulation is done using graph-oriented operations such as path, sub-graph, neighborhood, connectivity, centrality, and graph pattern. Rodriguez and Neubauer [RN10] defined graph database as a system optimized for efficient processing of densely interrelated dataset using graph data structure comprising of dots and lines. The dots are the vertices of the graph representing the entities while the lines are the edges for modeling the relationship among the entities.

Mathematically, a graph G is an ordered pair of two disjoint sets (V, E) where V is the vertices or node, and E is the edge [Bol13].The basic idea of graph data structure is to represent entities using the nodes of a graph and models the entities relationship with edges of the graph. Each node can have many attributes as possible and the connecting edges can have many attributes defining the relationship such as the cost, price, weight, temporal (time), spatial (distance) and ratings. In complex situations, two nodes may share several edges defining many interactions between the two nodes. Hence, in graph database, the entities are as important as their relationships. for instance, in Figure 2.6, the person entity has four relationships with the movie entity and has two recursive relationships with itself because a friend may not be a follower and a follower may not be a friend. Moreover, not all person will participate in each of the relationship. For instance, an actor may not be a

director through his or her acting career and a producer may never act in any movie while in another scenario, an actor may stop acting and start directing or producing movies. Thus, relationship among entities is dynamic and not static. Graph database models can be classified into four categories including Boolean, weighted, weighted multi-valued and dynamic model briefly described as follows:

- **Boolean graph database:** Captures Boolean interactions among entities. The node is a representation of entities, and an edge represents the presence of interactions between entities and absence of an edge between a pair of entity indicates the lack of interactions. Queries and pattern matching are simplified in Boolean graph database using the common Boolean logical operators (i.e., AND, OR, and NOT). For instance, let us consider situation when the spatial and temporal attributes are important. There are many of such situation in real-life day to day activities. For example, we [OLC21b], discussed the privacy protection of COVID-19 contact tracing, the trajectories of patient include the location visited before the infectious diseases' diagnosis and the time of visit. an area of interest is to model the spatio-temporal attributes of location and time of visit as relationships such that other people visiting the same location within a specific time are detected using simple Boolean logical operator.

- **Weighted graph database:** This involves assigning weights to the edges. By doing so, the degree of interactions between entities can be captured. The weights can be frequency such as frequent pattern mining of retail data, pattern of followers or the frequency of communication between two friends in a social network or the number of requests to a server from a specific IP address in telecommunication. Moreover, the weights can also be some other measures such as price, cost importance, temporal, spatial or risk metrics, which reveal the importance of these interactions. For example, in many real-life situations, frequent interactions may not lead to profitable or fruitful interactions. Hence, having the weights to measure importance or its equivalence is important.

- **Weighted multi-value graph database** Weighted Multi-value aims to cap-

ture multiple values such as frequency, importance, rates, cost, and risk factors on each edge. In terms of data to build such a graph database, it is well known fact that some real life domain such as social network, insurance and telecommunication modeling are based on relationship and grouping of individuals with common characteristics; for instance, for an employee insurance benefit registration, the information about dependents and other relative are collected in advance; in social network, the interesting pattern is to find most frequent followers and group them into specific category based on interest, a study [UHC20] groups computer professors into different categories based on specific topic they have participated, government usually classify citizen into different group for targeted benefit distribution and insurance companies capture those relations that are known in advance including the obvious connections among entities such as membership within the same family, membership attending the same school and membership working in the same company among other known relationship that are essential for managing risk.

- **Dynamic graph database**: As relations may change over time, which may be due to the availability of additional information and/or updates on existing information. This may also be due to changes in relations such as additions and removal of family members. In essence, dynamic graph database incorporates incremental changes among entities and reuse existing knowledge discovered from the previous instances of the graph database to reduce redundancy. Dynamic graph database operation includes nodes addition, node deletion, edge addition, edge deletion and incremental update. The dynamic graph operations are briefly described as follows:

  - Edge Addition: addition of new interactions and relationship such as marriage, merging of companies, and/or new partnerships without changing any existing data structure.
  - Edge Deletion: for instance, removal of existing interactions such as divorce, break-up of partnership, and/or division or split of companies.

- Node Addition such as addition of new family member or business partners; child adoption, new employee for group insurance, new student for student insurance among other additional information that are captured after the fact.
- Node Deletion: for example, death of people, employee's exit from a company or closure of a business.
- incremental update: Instead of reconstruction of graph database, incrementally update aims at keeping up with rapid changes to the graph database.

2.2.9 Knowledge Graph

Two common themes emerging from few definitions of knowledge graph in research literature today are big data integration and data reusability. For instance, Ehrlinger and Wöß [EW16] described Knowledge graph with five keywords— *information acquisition, integration, ontology, reasoner, and new knowledge* while Krause et al. [KHM+16] relates knowledge graph to high volume of data defining it as large networks repository for storing entities, entities relationships, entity property, and semantic types. Wang et al. Considered complex many-to-many entity relationship as a motivating factor for the emergence of graph models and defined knowledge graph as simply multi-relational graph model representing entities as nodes and relations with edges of the graph. For Wang et al., knowledge graph is simply an extension of graph model to knowledge representation in form of triplet facts such as entity-relationship-entity, head-relation-tail (h,r,t) or subject-predicate-object. Rather than defining knowledge graph, Paulheim [Pau17] described characteristics of knowledge graph as comprising of real-world entities, interrelationship of entities including arbitrary entities relationship and domain diversity.

In its simplest form, knowledge graph can be defined as an integrated framework for large scale knowledge acquisition from heterogeneous sources, knowledge representation and reuse based on graph architecture [OLC23]. This definition covers at least 4 Vs of big data:

- ***Value*** derived from knowledge reusability,
- ***Variety*** of data from heterogeneous sources,
- ***Visualization*** using graph representation, and
- ***Volume*** of acquisition and integration.

Knowledge graph has been studied for decades in different research fields and industries including mathematics [NH08], artificial intelligence (AI), Google [DGH+14], Yahoo [BCMT13], Microsoft[1]. The mathematics foundation of knowledge graph could be traced back to the introduction of knowledge graph theory by Hoede, a discrete mathematician at the university of Twente, Netherlands, in the 1980s. Other studies on knowledge graph between 1982 and 1993 include knowledge integration and structuring system (KISS) by Hoede's doctoral students [NH08], knowledge graph construction, representation and structure [Bak87], relationship extraction from knowledge graph [SdV88] and in addition, consistency and robustness of knowledge integration [Smi93]. Chen et al. [CWZ+20] traced the history of knowledge graph to the development of expert system in early 1970s, conceptual graph modeling [Sow84] and the emergence of semantic web models in 1980s for designing knowledge representation for meaningful exchange of information on the web using ontology for conceptual description of facts and reality through common vocabulary with the objective of ensuring that the world wide web data are machine readable and subsequently integration of those data may potentially become an easy task.

Knowledge graph open-source collaborative projects by academic communities to link web data in large scale, promote knowledge integration and reusability have evolved over time including Wikidata [VK14], YAGO *(Yet Another Great Ontology)* [PWS20], DBPedia [FBMR18], NELL *(Never Ending Language Learning)* [CBK+10] and BabelNet [NP12]. The summary of the following few academic knowledge graphs is presented in Table 2.1:

[1]https://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing/?msclkid=caf296abcf3211ec81d948a73a46ee15

- DBpedia: is described as a nucleus for web open data designed to provide large-scale machine-readable facts from Wikipedia data dump integrated with other open data web contents. DBpedia contains over 100 million facts, 5 million entities and over 100 million types without transitive closure [ABK+07].

- NELL *(Never Ending Language Learning)*: NELL is designed with the objective of continuous acquisition of knowledge from variety of data on the web and continuous learning of new knowledge. In essence, NELL supports large scale web data integration and reusability. NELL contains approximately 5 million entities and 432 thousand facts [Pau17, DGH+14]

- Wikidata: is a free collaborative knowledge base designed to manage the factual data from Wikipedia while retaining features such as plurality of data including uncertain facts, open editing, community control, accessibility, and continuous evolution. Wikidata has over 900 million facts, 78 million entities and 77 million types as at February 2020 [VK14].

- YAGO *(Yet Another Great Ontology)*: is a large-scale knowledge base designed from 78 million Wikidata items dump containing over 400 million facts combined with ontological backbone of WordNet [Mil98] and enforcing semantic constraints including domain, functional and cardinality constraints for logical consistency. The most recent version of YAGO has over 65 million entities, over 300 million facts and 70 million types [PWS20].

Table 2.1: Academic open knowledge graph projects

Project	Entities	Facts	Entity Types
DBpedia	5M	131M	114M
NELL	2M	432k	285
Wikidata	78M	900M	77M
Yago4	67M	343M	70M

The industry perspective of knowledge graph is not far different from the research description, for instance, Google in 2012 described its knowledge graph as a large-

scale technology to *leverage collective intelligence of the web* [Sin12] to provide all-inclusive breadth and depth search experience for users. In other word, big data integration of common knowledge and reusability can be seen as the motivating factors for Google knowledge graph. Other commercial knowledge graphs with the same theme, motivation and ideas have since evolve including Google knowledge vault [DGH+14] and Microsoft's Satori [GLH+18]. The summary of the following recent work in the industry is presented in Table 2.2:

- Google Knowledge Graph: A commercial large-scale web data integration designed to improve users search experience through iteration of knowledge acquisition, learning and reuse. Google knowledge graph contains 570 million entities, 18 billion facts, 1500 entity types and 35 thousand relation types[2].

- Google Knowledge Vault (KV): Knowledge vault has 3 main integrated components namely knowledge extractor, graph based prior and knowledge fusion system. The knowledge extractor is the knowledge acquisition layer retrieving data from the web including text, HTML tables, page structure and human annotated contents. The graph-based priors component uses the prior knowledge of the historical data in the existing content of the knowledge graph to determine the correctness of the new knowledge while the knowledge fusion is the data integration layer where addition decision is made for the incoming knowledge based on correctness score from the prior component. KV contains 45 million entities, over 270 million facts, 1100 entity type and 45 thousand relation types.

- Spark (Yahoo Knowledge Graph): is a semantic web search assistance tool aiming at integrating variety of web content data from heterogeneous sources including widely available data from Flickr, Wikipedia, X (formerly Twitter), Yahoo! Search, and other private data sources to improve search result ranking and recommendation for users. Spark consists of a distributed knowledge acquisition for data collection from different sources, knowledge extraction which

[2] $https://search.googleblog.com/2012/12/get-smarter-answers-from-knowledge_4.html$

include co-occurrence and popularity features implemented as a series of Hadoop MapReduce jobs, and a ranking system to decide the output relevant to user queries. Spark consists of over 3.4 million entities, 1.4 billion facts, 250 entity types and 800 relation types.

- Satori (Microsoft Knowledge Graph): is designed to understand the world around entities People, places and things and massive interactions among those entities to provide more relevant results to Microsoft Bing web search users' queries[1].

Table 2.2: Commercial knowledge graph

Product	Entities	Facts	Entity types	Relation types
Google Knowledge Graph	570M	18000M	1500	35000
Google Knowledge Vault	45M	270M	1100	45000
Yahoo Spark	3.4M	1400M	250	800

Knowledge graph is particular useful for describing and modeling telecommunication network, structures, functions, links and interactions among network devices and subscribers; for instance, Krinkin et al. [KVKZ21] developed hierarchical synthetic model for telecommunication network using statistics data on knowledge graph. In another study [YKZ20], hierarchical knowledge graph structure was designed to model a telecommunication TV cable network comprising of network topology, services, mobile and stationary devices, users access model, billing, and network data.

2.2.10 Data Warehouse

Data warehouse can be described as an integrated decision support system and repository for big data from heterogeneous sources. Several definitions and research on data warehouse architecture, design and challenges exist in literature with a common themes of data integration and decision support. For example, Inmon [Inm05] defined data warehouse using five keywords: *subject-oriented, integration, non-volatile, time-variant and decision support.* Kimball and Ross [KM00, RK13] described four distinct

components of a data warehouse to include the operation source system, the extract transform load (ETL), data presentation areas and business intelligence applications. The components are briefly discussed as follows:

- The operation source systems capture business transactions and may or may not share common data. for instance, an organization may have retail application and a marketing application as legacy systems that are not integrated and do not share data. In such cases, multi-data entries are common rather than sharing the common data in an integrated environment, clerks enter the data across multiple systems.

- The extract transform load (ETL) system involves extraction of data from the disparate operation source systems, transformation which may includes data cleaning and resolving missing values or data type inconsistency, and finally loading of the data into the data presentation areas.

- The data presentation area encompasses the data warehouse environment where data are modeled, stored and accessed through queries or using the business intelligence application.

- The business intelligence applications refer to all simple or complex applications leverage the data presentation areas by querying the data warehouse to provide business intelligent reports for decision makers.

Two school of thoughts exist for data warehouse architecture classification, first notable forms of classification is layered-oriented which can be single layer, two-layers or three-layers architecture. The single layer has only the operation sources system and the data warehouse can be seen as materialized views created from multiple sources. A major task in single layer data warehouse configuration is view materialization which is a process of selecting the most efficient relevant views to be included in the repository for answering all queries of interests and at the same time minimize materialized views maintenance cost. The obvious issue is that it is practically impossible to predetermine all queries of interests before designing a data warehouse.

Two-layers architecture simply separate the heterogeneous data sources from the data warehouse using the ETL as a staging layer for data transformation while a three-layers architecture aims at integrating the transformed data from the heterogeneous sources before loading into the data warehouse. In essence, both two-layers and three layers can be seen as a multi-layer architecture recognizing the significance of the data integration through the extract-transform-load (ETL) system in big data management.

The other school of thought has five classifications which are essentially a multi-layered architecture briefly described as follows:

- Hub and spoke architecture, which was introduced by Inmon et al. [Inm05]. The notion of enterprise data warehouse comprising of the data source, the ETL system, a mandatory normalized data warehouse and several functional data marts extracted from the data warehouse for decision support system. User access is mostly restricted to the functional data marts. The normalization requirement adds to the design complexity of this approach.

- Independent data mart architecture, which refers to non-integrated data sources designed for different functional areas within the same organization. An obvious drawback of this approach is non-sharing of common data that may have relevant hidden pattern for decision making.

- Bus architecture, which was introduced by Kimball and Merz [KM00, RK13]. They added the logical integration to the independent data marts architecture using business requirements to relate and create a multi-dimension model sharing common design guidelines and principles.

- Centralized architecture, which was a further refinement of Inmon et al.'s hub and spoke architecture to a centralized architecture by eliminating the previous data mart layer and thus achieved an enterprise-wide integration of big data sharing common logical and physical designs.

- Federated architecture, which aims at ensuring sharing of common data across multiple related organizations or supply chains using different interoperability

framework such as distributed queries and metadata with little to no change in existing data warehouse infrastructure.

2.2.11 Data Lake

The growing varieties and heterogeneous sources of collecting structured, semi-structured and unstructured data in addition to up front cost of designing and maintaining rigid data warehouse motivated the development of a new high-capacity repository structure for raw big data known as *data lake (DL)*. A common theme in data lake literature is the notion of centrality and integration of data to enable accessibility, collaboration and analysis of big data siloed in several repository including databases, file system, and data warehouse.

The notion of *data lake* emerges from industrial effort to solving the challenges of storing disparate data types from heterogeneous sources and the increasing volume of data. Chessesel et al. [TSRC15] and Terrizanno et al. [CSN+14] of IBM research defined data lake as a centralized repository for large structured and unstructured data described by metadata organized into identifiable set and available on demand.

There exist few definitions of data lake in research literature. Fang [Fan15] defined data lake as low-cost central repository for capturing, refinement, archiving and exploration of enterprise data. Madera and Laurent [ML16] defined data lake relative to data accessibility as a logical view of all data sources available to data scientist and statisticians, however, the study did not account for data gravity, governance and other community of users, additionally, the study limited the technology to Hadoop solution. Whereas, other studies of data lakes focused on data integration, for instance, Walker and Alrehamy [WA15] considered integration of personal data in their description of data lake and Hai et al. [HGQ16] defined data lake as big data repository for on-demand integration. Data quality related definition is provided by Maccioni and Torlone [MT17] defining data lake as large scale loosely collection of data with no requirement for data quality.

2.2.11.1 Data Lake Architecture

Few data lake architecture exists in research literature, The three common approaches to data lake design are:

- Flat architecture, which is a single zone schema-less architecture enabling storage of diverse data type from heterogeneous source with no option of data processing or plug-in to extract useful pattern from the vast amount of data in the data lake. Inmon [Inm16] described the flat architecture as a one-way garbage dump of data with no future use. In other word, this type of architecture does not support accessibility and reusability. Thus, *flat architecture can be defined as low-cost repository for large non-reusable big data from heterogeneous sources.*, An obvious disadvantage of the flat architecture aside non-reusability is that there is no support for analytical data processing for finding insightful pattern for business decision making.

- Pond architecture, of which the proponents were motivated by the challenges arising from the flat architecture inability to allow processing of raw data for further use in the business value chain. Five ponds are recognized for data lake pond architecture starting from the raw pond where all data are ingested into the data lake and are transformed according to three general characteristics of the subsequent ponds which may be analogue for measurement data, application pond for application generated data and textual pond for unstructured data; the last flow for the pond architecture involves archiving of old non-use data from the analogue, application or textual pond into the archival pond. The architecture is presented in Figure 2.3 and the five ponds are described as follows:

  1. Raw data pond is a repository for raw data of diverse data type from heterogeneous sources. A conditional processing is executed to transform the data to the appropriate subsequent pond.
  2. Analogue data pond is a repository for non-application data. Analogue data are repetitive voluminous data generated by machines or other auto-

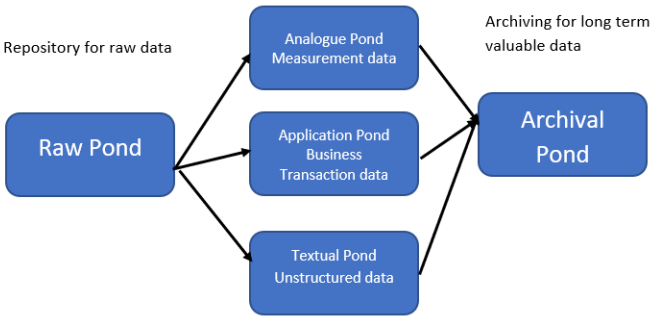


Figure 2.3: Data lake pond architecture

mated devices such as log files data, CPU usage, weight, height and blood pressure of a patient and other physical measurement. Data transformation in form of cleaning and reduction of voluminous data to manageable size are executed to enable meaningful data analysis.

3. Application data pond is a repository for storing data resulting from execution of one or more business transaction from different application without integration.
4. Textual data pond is defined as a repository for unstructured textual data with no specific pattern. Data of this format often needs conditional processing to identify the *context and association or relationship* .
5. Archival data pond is simply a long-term repository for valuable unused data from analogue, application or textual pond.

- Zone Architecture, unlike pond architecture, which segregates data lake using combination of data type, usage and life-cycle status, zone architecture has no standard structure and different researchers have defined zoning in different ways. For instance, John and Misra [JM17] proposed three-zones Lambda architecture having: (a) batch for maintaining and processing of raw data in batches; (b) speed or real-time layer where real-time analysis of business requirements is done through merging of batch views and real-time data; and (c) serving layer provides the views created in batch and speed zones to users. The

Lambda architecture aimed at achieving three objectives; first, fault-tolerance for hardware, software, and human errors to prevent data loss and data corruption; secondly, immutable data storage allowing raw data insertion without deletion to reduce human errors and lastly, re-computation of new business requirements from raw data on demand. A major drawback of Lambda architecture is the cost of maintaining a sync between the batch and speed layer as well as the effort required to creating views in both layers and integrating real-time data with batch view in the speed layer. Meanwhile, other researchers [Sha18, GGH+19] have described data lakes using level of data refinement. For instance, a six zones architecture was described [Sha18] to consist of the *transient loading zone* as the entry point for raw data and basic quality checks, the *raw data zone* takes as input, data from transient zone for cleaning and standardization, then the *trusted zone* accepts the standardized data from the raw data zone, next is the *discovery sandbox zone* for data scientist performing data analysis, data wrangling or creating machine learning model, the fifth zone is the user *consumption zone* where self service operation such as filtering, applying user-defined constraint and other non-technical analysis through dashboard tools are performed and finally, there is a *governance zone* for managing big data quality and security. The six-zone architecture seems too clumsy and complex to manage because there is an overhead cost of maintaining multiple copies of the data move from one zone to another. In fact, the pond architecture may be considered as a variant of zone architecture having five zones since both pond and zone architecture involve data movement from one area of the data lake to another within big data management life cycle. Hence, both the pond and zone architecture may suffer from data loss issue due to data cleaning, the pond architecture for instance, delete the raw data once it is refined and transferred to the analogue, application, or textual zone.

2.2.11.2 Data Lake Issues and Challenges

Data lake issues and challenges have been identified and discussed in research literature, for example, Giebler et al. [GGH+19] examined current state and challenges of data lake including architecture framework, governance, and design strategy issues. Terrizzano et al. in their discussion of data lake issues likened the issues associated with raw big data collection and interpretability to a journey from the wild to the lake [TSRC15]. Though, there exist numerous values in centralizing several siloed data in many businesses and organization including data integration and ad-hoc self-serve data analysis and reporting, there are several issues and challenges associated with the design and maintenance of a data lake, few issues were identified [CLSO22] as listed below:

- Big data lake architecture: Designing and implementing big data lakes may be a daunting task if the aim is effectiveness and efficiency. In this sense, one key aspect is the architectural design of the big data lake itself. Indeed, the architectural design of a data lake has a big impact on its performance. Hence, an important research open issue is the architectural standard for their design, which may be smartly accomplished through inspiration from early data warehousing systems designs. Several studies have proposed zone architecture which could be likened to the layered-oriented architecture approach of data warehouse but, without schema. In other word, since the common themes for data warehouse and data lakes revolve around data heterogeneity, centrality, and reuse; the architecture framework of data warehouse could be adapted to a schema-less layered data lake architecture.

- Data provisioning paradigms for big data lakes: Data provisioning is a fundamental and essential process of data warehousing systems. It imposes how input data sources are imported into the data storage layer of the target data warehousing system while considering the ETL (Extraction, Transformation, Loading) process. The question is, what is a suitable data provisioning process for big data lakes? The 3V input nature of big data makes it challenging to be applied as-is, as highlighted by recent studies including [BBA17]. Novel

solutions may reveal themselves to be crucial. For instance, sampling-based techniques seem to be worthy approaches to employ.

- Query-rewriting paradigms for big data lakes: A big data lake stores big data repositories in its core storage layer where these repositories remain in their original form until they are user-prompted for example through a query. The "lake" has the role of mapping schema of the ingested data and, most importantly, has to include query rewriting mechanisms in its core module. Designing effective and efficient query rewriting techniques for the core layer of a big data lake is, undoubtedly, a challenging problem to deal with in future big data lake research.

- Modern metadata managers for big data lakes: Metadata management within big data lakes are crucial for the good functioning of the lake. An essential feature to be included within a big data lake is big data exploration. Given that standard big data lake architectures are naturally intricate, effective metadata management design and implementation are key to supporting effective and efficient big data exploration. Classical approaches are not sufficiently effective on modern big data application infrastructures, thus, an innovative, modern metadata management design and implementation paradigms should be set.

- Big data lakes and machine learning tools: Big data lakes intrinsically support machine learning tools. The inclusion of such tools will enable value inference from core big data repositories, in the purpose of discovering knowledge and support decision making. Over the years, many machine learning tools have been proposed in the literature. Sadly, they adopt a trivial and raw (input) data model, which is unable to close the gap for modern features of big data repositories and hence falling as a non-suitable model. Consequently, designing effective and efficient machine learning tools remains one of the most relevant challenges to face, despite their conclusive results in a wide range of real-life big data applications, "fluctuating" from healthcare analytics to social evolution analysis tools, as well as in smart city applications and e-science solutions, to

list a few.

- Big data governance methodologies via big data lakes: Data governance is defined as a set of models, methodologies and paradigms supporting what is known as data-driven, "decision-making-enabling", digital society, i.e. a futuristic vision in which the main societal processes should be driven by analysis of historical and current data. Indeed, nowadays, a clear case that adheres to this vision is plain concrete: the COVID-19 pandemic outbreak and the policies that countries are applying to decelerate the disease spread of the epidemic, being these policies totally driven by the daily analysis of data of the outbreak evolution. The question to be asked is the following: should and could big data lakes be the answer to support a real big data governance for next-generation societal challenges? This question raises a crucial issue that future research endeavors should tackle.

- Privacy-preserving big data lakes: In opposition to data warehousing architectures, where the privacy-preserving data approaches problem have been "deficiently" dealt with, privacy-preservation methods for big data lakes should be enhanced since, big data applications inputs, in the latter case, usually are of sensitive data types such as social data, censor data, telecommunication data, biomedical data, healthcare data, among others. Hence, there is a need to achieve privacy-preserving for big data lakes. In this sense, big data lake architectures should have all of their core procedures including data input operation, query rewriting, metadata management and other functions designed and implemented according to privacy-preservation data management paradigms which may constitute the most relevant challenge for future-generation big data lakes.

- Emerging big data lake applications: It should be noted that big data lakes are the typical architectures where real-life applications could lead to improvements of architectural components via continuous iterative tuning and updates ordered by real-life metrics including temporal complexity of big data access and query procedures, accuracy of big data analytics front-end tools and usability. Hence,

modern application scenarios such as smart cities, e-government, healthcare management, energy consumption systems, will be big data lake architectures that will adapt depending on the application domain. The research challenge consists in learning the lessons, from relevant and trustworthy case studies of foundations trials of big data lake architectures, from practice to theory.

Other data concepts have emerged recently focusing more on data analytic infrastructure and federated governance notably is the data mesh and data fabrics.

2.2.12 Data Mesh

Data mesh [Deh22] coined by Dehghani in 2022 as a socio-technical concepts aiming at enabling data democratization using self-service data infrastructure platform, federated data governance and domain-oriented decentralised data ownership approach. Thus, data mesh is based on four principles [BKKH23] briefly defined as follows:

1. *Data-as-a-product*, which refers to product-thinking approach to data such that end-to-end responsibility around data life-cycle management is well-defined and structured resulting in the emergence of new roles such as data product owner, data product developer and data product customers. In addition, Zhamak described six usability characteristics of data product to include Discoverable, Addressable, Trustworthy, Self-describing, Interoperable and Secure (DATSIS).

2. *Domain-oriented data ownership*, which refers to universal availability of data within a domain that is interoperable across organization based on the concept of domain-oriented decentralized data ownership.

3. *Self service data infrastructure platform*, which refers to an autonomous data infrastructure including tools and interfaces aiming at reducing reliance on highly skilled technical expertise by dedicating a domain team to the production, support and maintenance of data products interoperable to multiple users across the organization while aligning the domain to the data source and data products consumption. [Deh22, MCS22b]

4. *Federated governance*, which refers to interoperable standard and policies across multiple domains within an organization.

Data mesh is a relatively new concept with very few academic research interests and thus lack scientific foundation [MCS22a]. Machado et al. [MCS22a] presented the technology architecture for the implementation of a data mesh and discussed the features, concepts, principles and implementation approach highlighting two use cases from the industry—Netflix streaming platform and Zalando online retailer [MCS22b]. In addition, the authors discussed the conceptual architecture towards achieving a decentralized data mesh [MCS21].

Despite the benefits of the data mesh of organizing organization team in form of data-driven domain groups responsible for data quality, security, analytics and governance, the concept of data mesh lack clarity on issues such as shift in roles and responsibilities, architectural framework for the data products, difficulty in enforcing federated governance across heterogeneous domains [BKKH23] and privacy concerns across heterogeneous data mesh compositions [PKM22]. Data mesh lacks scientific foundation [MCS22a] and the principles are more related to data-driven domain-oriented organization of teams to manage the production, support, maintenance and governance of data in a distributed environment and hence the conceptual model for data storage is completely lacking.

2.2.13 Data Fabrics

While data mesh promotes decentralization, data fabrics advocates for centralization of data from heterogeneous sources using cataloging to ensure data are discoverable across the organization. Similar to data mesh, data fabric evolves from the industry and hence, most definitions are found in business blogs including bmc[3],

[3]https://www.bmc.com/blogs/data-fabric/

dataversity[4], talend[5], data management blog [6] and cloudera[7], [8]. Data fabrics is commonly defined by the business blogs as an integrated infrastructure platform for data management enabling cross sharing of data with five layers comprising of the following:

1. Data Ingestion: collection of data from heterogeneous sources
2. Data Orchestration includes tasks such as data cleaning, data curating and transformation.
3. Data Analytic tools
4. Data Governance: Centralized policies for data access and security
5. Data consumption: An interface enabling data access to various users.

Thus, data fabrics lacks architectural framework for defining data storage in term of variety and veracity.

Very few definitions of data fabric exist in research literature. Alvord et al. [ALDC22] opined that data fabric itself may not be appropriate for big data due to its limitation of handling only structure data and the existence of non-integrated distributed analytic tools, and defined big data fabric by combining big data concept with data fabric as an integrated platform for heterogeneous data collection and processing using data lakes, Hadoop [9] and Apache Spark [10] technologies enabling big data large scale analytics and proposed big data fabric architecture comprising of the following five layers:

1. Data Ingestion: collection of data from heterogeneous sources

[4]https://www.dataversity.net/secrets-utilizing-data-fabric/
[5]https://www.talend.com/resources/what-is-data-fabric/
[6]https://www.datamanagementblog.com/key-to-weaving-big-data-fabric/
[7]https://blog.cloudera.com/data-360/how-a-big-data-fabric-can-transform-your-data-architecture/
[8]https://tdwi.org/articles/2018/06/20/ta-all-data-fabrics-for-big-data.aspx
[9]https://hadoop.apache.org/
[10]https://spark.apache.org/

2. Data management including governance, security, processes, policies and data quality.

3. Data Orchestration includes tasks such as data cleaning, data curating and transformation.

4. Data Discovery is a layer for the analysts to discover patterns and provide reports to users.

5. Data Access: refers to an interface for interacting with the users.

2.3 Comparisons among Database Models

Before comparing different databases in the following sections, the historical overview and the motivating theories for the evolving database models are presented in Table 2.3.

Table 2.3: History and theories of database models

Database model	Year	Theory	Data structure
Hierarchical model	1960s	Hierarchical	Tree
Network model	1960s	Network	Pointer, record
Relational model	1970s	Set, predicate logic, relation	Relation
Entity-relation (ER) model	1970s-1980s	Set, relation, logic, lattice	Relation
Object-oriented database model	1980s-1990s	Object state, behavior	Object
Semantic database model	1990s-2000s	Knowledge representation, graph	Tree
Semi-structured model	1990s-2000s	Graph	Tree
Graph database model	1980s-2020s	Graph	Graph

2.3.1 Data Warehouse vs. Data Lake

The common motivating issue for the introduction of data warehouse and data lake is the need to store and integrate disparate data from heterogeneous sources in a central repository. Data lake does more, by providing a promising solution to big data varieties issues of storing structured, semi-structure and unstructured data without the need to design schema for the repository. In addition, data warehouse is designed with the notion of one-size-fit-all, a situation that is very rare in real life big data applications. Other challenges of using data warehouse include:

1. Data warehouse is characterized with well-defined schema, data structure, complex pre-processing including rigid process of extraction, transformation and loading (ETL) data to ensure the data conforms to the architectural design of the data warehouse in support data life cycle management. There are likely potentially useful data loss in the pre-processing and ETL tasks.

2. Using data lake has cost benefit advantages, for instance, the upfront cost associated with building a data warehouse including cost of data modeling, transformation and integration may be deferred or completely avoided.

3. The schema-less approach of data lake provides opportunity for storing varieties of data types in different formats such as graph, audio, video and other complex proprietary format files (e.g., SEG-Y and ATX files, which normally would not be stored in the data warehouse).

4. The integrity of the raw data is preserved in a data lake unlike the rigid transformation requirement of data warehouse in which case the data are pre-processed to conform to the structure of the underlying data warehouse data architecture.

5. With regards to Users of the two scalable repositories, both Data warehouse and data lake serve the diverse community of users—including data scientist, developers, machine learning engineers, data engineers and business analyst.

6. Data warehouse and data lake are essentially data repository for storing big data from heterogeneous sources.

7. It is worth noting that a data warehouse can be a data source for a data lake as the former contains structured data and a data lake can also be a data source for a data warehouse because the semi-structure and unstructured data in a data lake can be pre-processed, extract, transform and load into a Data warehouse.

2.3.2 Relational Database vs. Data Warehouse

Relational database is optimized for Online Transaction Processing (OLTP) supporting common *insert, update and delete* operations as well as information retrieval

using the SQL language. Whereas, data warehouse is designed to support online analytical processing (OLAP), decision support systems(DSS) and data mining. OLAP is an essentially tool for the analysis of complex data using distributed computing capabilities and huge storage capability of a data warehouse. DSS is built to support decision makers with high level complex data while data mining is concerned with finding previously unknown pattern from the huge volume of data stored in the data warehouse.

Another obvious difference between relational database and data warehouse is the volume of data. Data warehouse is designed using multidimensional model for storing very large volume of data from heterogeneous sources including data from multiple relational databases.

Data warehouse generally supports time series and trend analysis because it has more historical data than what is available in most relational database. Data warehouse is a non-volatile storage system requiring periodic update (also known as refresh) unlike relational database where insertion and update are regular frequent operations. The data warehouse has a well-defined refresh policy such as incremental update in which case new data are added to the existing data. The smallest unit of data in most relational database is a unit of transaction, whereas data in data warehouse are more coarse-grained or aggregated in nature.

2.3.3 Relational Database vs. Graph Database

Relational databases have existed for several decades and widely used for the storage and retrieval of structured data using a common language known as Structured Query Language (SQL) across multiple types of relational database management system including Microsoft SQL and ORACLE. Despite the wide adoption of relational database, it has some limitations including the requirement for rigid schema, inefficient handling of information retrieval in situation requiring several JOIN operations in query processing and also, managing dynamic and complex relationship among entities may be operationally expensive without rebuilding the schema. Thus, the interest in graph database has evolved to solve the challenges associated with man-

aging dynamic and complex relationship among entities in different domain such as biological, medical, social network, finance, insurance, and telecommunication data. For instance, as a result of inefficiency in query processing by the legacy system built on relational database, Fabregat et al, [FKV+18] developed tool for migrating an open data for public sharing of complex bio-molecular data from relational database to graph database. Srivastava and Singh [SS23] designed a fraud detection algorithm using a distributed graph database and node ranking to identify fraud by minimizing the least influential node; Prusti et al, [PDR21] applied graph database to credit card transaction data to detect fraudulent financial transactions.

Ease of integration in a schema-less dynamic graph database is another major benefit of using graph database over relational database. dynamic addition of new relationship without rebuilding schema is an interesting feature that has evolved over time for big data integration. For instance, due to the fixed schema limitation of relational database in modeling heterogeneous data source, Henkel et al. [HWW15] applied graph database concept to support modeling and integration of heterogeneous biology data such that the computational models, the bio-ontology semantic annotation [DSD+11], the metadata and the simulation experiment are effectively combined as a network represented by graph data structure. In the field of biochemistry, there exits several public data portals including protein sequencing database [Con15], genes and genomes database [KG00] among others. Though, the publicly available data sources promote data reusability, integrating the data from the heterogeneous sources is a major challenge. Hence, Swainston et al. [SBC+17] designed a graph database known as biochem4j to integrate chemical reaction, enzymes and taxonomic data from several biochemistry heterogeneous data sources using several interconnected nodes to represent specific chemical entities and edges to define the chemical interaction among the entities, thus allowing further analysis of the biochemical data through complex queries and pattern visualization.

Visualization is another distinct advantage of graph database over relational database. Relational database is table-like structure containing rows and columns; to visualize patterns and trends in big data stored in such relational database beyond tabular presentation, another business intelligence or visualization tool is required.

Whereas graph database combined storage and visualization using network of nodes and edges. A recent use is demonstrated by Mao et al. [MYZ+21] for the analysis of COVID-19 epidemic contact tracing data, the study built a graph network of relationship among highly infected individuals and the public for contact tracing while Ubaru et al. [UHC20] applied dynamic graph model to identify individual exposure for prescribing appropriate testing in a minimal testing capacity scenarios and testing budget constraint.

Over time, some other technologies such as extensible markup language (XML), Hypertext markup language (HTML), JavaScript object notation (JSON) and Not-only-SQL "NoSQL" have evolved in order to overcome the challenges of handling semi-structured and unstructured data types of big data. However, these new technologies focus more on data structure rather than entities complex relationship management. Thus, graph database has become an essential tool for the storage of big data as well as management of complex dynamic relationships among entities most importantly when such entities have underlying hierarchical many-to-many relationships that are difficult to model using relational database. As an example, let us consider the common movie database in relational database shown in Figure 2.4 where there are four entities namely: Movies, Actor, Producer and Director. There are six relationships with varying cardinality constraints specifying the maximum number of relationships an entity can participate in. There are two one-to-one cardinality relationships, one between actor and director and another one between actor and producer; meaning that an actor can be a producer and a director. The one-to-one relationship is very easy to model in relational database by simply adding the primary key of one entity as a foreign key in the other participating entity. However, if we examine the many-to-many cardinality constraints between actor and movie and the other between producer and movie, then we have a complex situation to model because a movie can have many producers and additionally, some actors can act in many movies. Thus, the design of the normalized relational database resulted in creation of two additional relationship tables in the schema design to eliminate redundancy as shown in Figure 2.5.

Additionally, adding social network to the Movie database creates some further

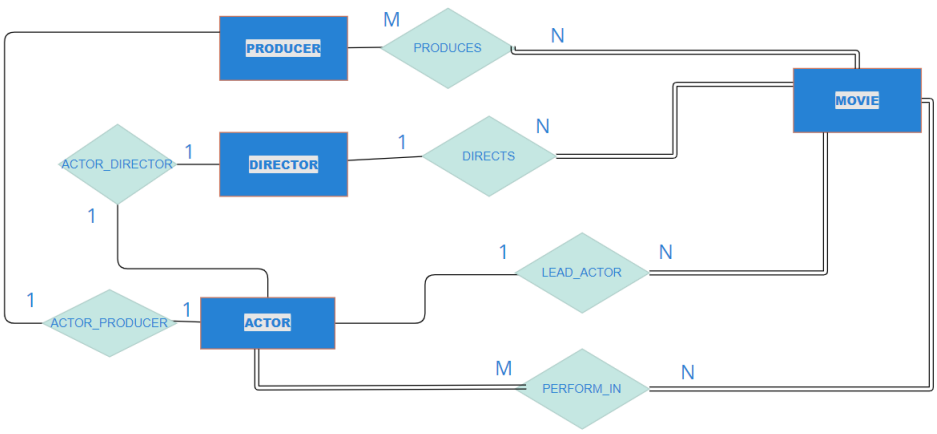


Figure 2.4: Movie entity relationship diagram

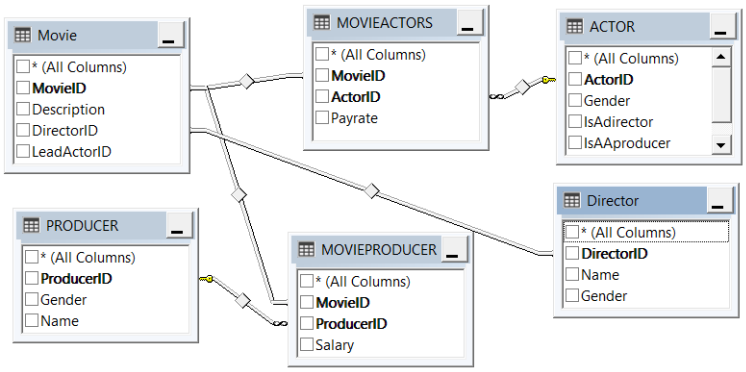


Figure 2.5: Movie physical design

challenging questions including:

- Who is the actor, director, or producer followers on social network?
- What is quantitative weight of an actor rating?
- Is there a director who is a friend to a particular actor and whose comments generates more followers for the actors?
- What are the following patterns?

Thus, there are lot of questions relating to complex socio-interaction in today's world that are difficult to model in relational database. Hence, graph database has essen-

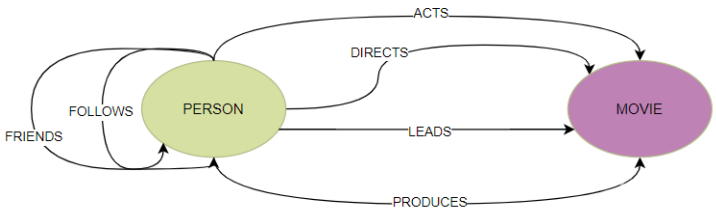


Figure 2.6: Movie graph database model

tially become an important tool for modeling complex many-to-many societal scenarios and issues in social network, healthcare, telecommunication among others. For instance, the graph database model for the same Movie database is shown in Figure 2.6 where we have two nodes and six edges. The node simply captures two entities: Person and Movie because an actor, a director, a producer, or lead actor are actually an instance or a subclass of a person. Edges represent relationships and we can add as many edges as possible to model the complex many-to-many social interaction among entities. For instance, an actor can be a friend to a director, a director may not necessary be a friend to the actor but may be following the actor on Instagram. This is a typical case of Boolean graph database where the Boolean logical operators (AND, OR, NOT) become useful in querying the existence and non-existence as well as frequent and infrequent social interaction among participating relationship entities. The challenge of mining the social interaction becomes obvious in situation when $A \leftarrow D$ is not the same as $D \rightarrow A$ in which case, the actor following a director on Instagram, but the director is not following the actor. Thus, we need to capture a third scenario to specifically model the case of $A \leftrightarrow D$ when the actor and the director are actually following each other. Hence, graph database is essentially a useful tool for social network modeling. Some examples of studies that have utilized graph database in the past include:

- Ho et al. [HWL12] designed a distributed graph database system to manage social network data leveraging graph partitioning optimizations techniques in the design of the distributed database graph storage system for efficient social computing,

- Nicoara et al. [NKDC15] designed a repartitioner in support of social network data partitioning over multiple graph databases.
- Soussi et al. [SAB10] proposed a framework for transforming relational database to graph database to support social network pattern extraction.
- Mezzanzanica et al. [MMC+18] designed a graph database containing over five million nodes and over twenty-four million relationships to model social interaction among computer scientist including their publications, profile and social media interaction on specific topics using multi-graphs and word embedding techniques.

Since there is no requirement for designing any rigid schema for graph database; the non-trivial task of redesigning database schema due to change in relationship or business rules can be avoided.

2.3.4 Relational Database vs. Data Warehouse vs. Data Lake vs. Graph Database

In summary, graph database is a useful tool for modeling dynamic complex relations such as addition of new relationship such as new follower, new child in the family, new partner or any other newer relationship and deletion of obsolete relationship in such cases as director un-following an actor. Since there is no requirement for designing any rigid schema for graph database; the non-trivial task of redesigning database schema due to change in relationship or business rules can be avoided.

A summary of comparison of relational database, data warehousing, data lakes and graph database is presented in Table 2.4.

2.4 Open Data Principle

The concept of open data is built around four primary principles: Findable, Accessible, Interoperable and Reusable (FAIR).

Table 2.4: Database comparison

Criteria	Relational DB	Data warehouse	Data lakes	Graph DB
Data Types	Structured		Structured, semi-structured, unstructured	Semi-structured
Data representation	Row-column		Data "as is"	Nodes-edges graph
Architecture	Highly structured		Unstructured	Semi-structured
Data processing optimization	Optimized for non-complex online transaction processing (OLTP)	Optimized to support online analytical processing (OLAP)	Optimized for OLTP, OLAP, complex dynamic modeling	Optimized for complex and dynamic entity relationship modeling
Agility	Less agile		Agile	
Schema Requirement	Schema required		Schema optional	
Pre-Processing Requirement	Complex			Simple

2.4.1 The FAIR Principles

The purpose of the FAIR guiding principles is to provide fundamental domain-independent guidelines with regards to the open data publishing, data sharing, optimal use and reuse by human and machine. Though, the four primary elements constituting the FAIR principles are related, they are independent and separable [WDA+16] and the interpretation varies across different domains [JdJ+20]. The four components of the FAIR principles are discussed as follows:

- **Findable:** An Open data object is findable if it is uniquely and persistently identifiable. A typical approach of ensuring that an open data object is findable is the use of digital object identifier (DOI) which is a permanent identifier for an article or document on the internet. Juty et al. in [JWS+20] discussed other options for implementing the findable principle to include uniform resource identifiers (URL), archival resource key (ARK) and persistent uniform resource locators (PURL). In addition, a findable data object has a metadata associated with it. A metadata commonly refers to as "data about data" is a description of the data object including its purpose, contents, authors, date published, update frequency, terms and condition of use. Indexing, an important concept in database is the last part of findable principle; the data object must be indexed in a searchable resource such as schema.org and the data catalog vocabulary

(DCAT). The searchable resources provide the infrastructure for discovering the data object using the unique identifier or any attributes within the metadata [JdJ+20]. The findable principle is the foundation for the other three primary principles.

- **Accessible:** A data object is accessible if there is an open, free, predictable, and universal communication protocol explicitly defined to retrieve the data and its metadata by human and machine. In addition, the authorization and authentication procedure must be clearly defined for any access restricted data object. A common standard protocol is the hypertext transfer protocol (HTTP). The second part of the accessible principle is the continuous availability of the metadata when the data object is no more available due to legal requirement to destroy some data after certain period. Thus, implementing the accessible principle ensures that the history of a data object is preserved.

- **Interoperable:** The purpose of the interoperability is to achieve a common understanding of digital data object using a common language and vocabularies within the same domain. A typical common language that ensures interoperability is the Resource Description Framework (RDF).

- **Reusable:** The terms, condition and usage license of data object must be clearly defined to promote reusability. The detail of WHAT, WHO, WHEN, WHERE and HOW associated with the data object must be clearly defined in metadata.

The summary for the FAIR principles is presented in Table 2.5.

Table 2.5: Open data FAIR principles

Principle		Context Description
Findable	F1	Data Object has unique and persistent Identifier
	F2	Data object has rich metadata
	F3	Metadata are linked to the data object it described
	F4	Data object is indexed in searchable resources
Accessible	A1	Data object is retrievable using open and free standard communication protocol
	A2	The protocol provides authorization and authentication procedure when necessary
	A3	Metadata are accessible when the data object is no longer available
Interoperable	I1	Use accessible language for common knowledge representation
	I2	Data object and its metadata use common vocabularies within a specific domain
	I3	Data object includes qualified reference to other digital data object
Reusable	R1	Data object is richly defined with clear plurality of accurate and relevant attributes
	R2	Data object has a clearly defined usage license
	R3	The data object is associated with detail provenance.
	R4	Data object meets domain relevant standard.

2.5 Big Data Privacy Preserving Models

Due to privacy concerns relating to big data management tasks such as data publishing, data mining and data distribution; several privacy preserving models have been designed by applying one or combination of generalization, perturbation, anatomy or suppression to sensitive and / or quasi-identifiers in a given dataset. A sensitive identifier is defined as personal attributes of an individual such a names and social insurance number while a Quasi-identifier is an attribute or set of attributes that though not explicitly personal but can be linked to other information to identify a person, an example is the postal code or Zip code. Note that this research has previously discussed how a researcher was able to identify a state governor in the United State by combining US zip code with other publicly available data in Section 1.2. The four techniques for privacy preserving models are discussed as follows:

- Generalization, which is similar to the concept of generalization described by Navathe [Nav92] where employee entity is a generalization of all possible professional specialization in a company such as Engineers, Accountants, Program-

mers, Nurses and Doctors. Another example in this technique is the categorization of numeric values, for instance, age could be put into different equivalence class such as 5-10, 11-15 and so on.

- Anatomy, which was proposed by Xiao and Tao [XT06] as a variant of l-diversity, the technique creates two tables, one for sensitive identifiers and the second one for quasi identifiers to prevent linkage of the sensitive and quasi identifiers.

- Perturbation, which involves replacing the original data with a statistical identical data using (a) Deep learning model known as the generative adversarial network (GAN) [CWD+18] to learn from the original dataset and generate a privacy preserved identical data as output. Another model in this category is data swapping in which sensitive data are exchanged between two or more entities within the same dataset. Lastly, adding noise to data which may be additive or multiplicative noise is also an example of perturbation models.

- Suppression, which refers to the removal of an attribute or set of attributes that are considered to be sensitive or quasi-identifiers from a dataset to preserve privacy and identity of the individual record owner.

2.5.1 Privacy Preserving Generalization and Suppression Models

Models that combine generalization and suppression techniques for preserving privacy in data publishing task include k-anonymity, l-diversity, and t-closeness models:

- *k-anonymity* model is designed on the concept of equivalence class, in which at least k records are indistinguishable. However, the model does not consider sensitive identifiers, and the assumption that individual has only one record in the dataset may not always hold. Thus, issues arise when several or all records within an equivalence class belong to the same individual. In such a situation, disclosure is possible.

- *l-diversity* model [MKGV07] was introduced to resolve the non-inclusion of sensitive identifier in the equivalent class constructed using k-anonymity model. l-diversity essentially add diversity to the member of equivalence class records by including sensitive identifiers. l-diversity principle assumes that an equivalence class is *l-diverse* if it has at least *l-well represented* sensitive identifier. Obviously, the term "well represented" is subjective, without concrete definition or measure. Hence, there exists three further classifications of l-diversity—namely distinct l-diversity, entropy l-diversity, and recursive (c, l)-diversity. For *distinct l-diversity*, there are at least l distinct sensitive attributes in each equivalence class, which is essentially a case $k = l$. As a stronger form of l-diversity, *entropy l-diversity* describes the distribution of sensitive identifiers such that an equivalence class E has $Entropy(E) > log(l)$. *Recursive* (c, l)*-diversity* ensures there is balance in the representation of frequent and infrequent sensitive identifiers in a given equivalence class. Despite the diversity of sensitive identifier in equivalence class, skewed distribution of attributes and limitation of adversary knowledge [LLV07] may increase the risk of disclosure using l-diversity and its variants.

- *t-closeness* model [LLV07] was designed with emphasis on global knowledge distribution to ensure that the distribution of the sensitive identifiers in the equivalence class is close (i.e., similar) to the distribution of the sensitive identifiers in the original dataset. The threshold t is a distance metric defined such that $Dist(Q, P) \leq t$ where Q is the distribution of the sensitive identifier in the original dataset and P is the distribution of the same sensitive identifier in the equivalence class.

To recap, k-anonymity, l-diversity and t-closeness were designed and applicable mainly to relational databases. Due to high dimensionality and sparseness nature of spatio-temporal data, the computation cost of these models may be exponential. For instance, the complexity of k-anonymity on relational data is exponential to the size of quasi-identifier 2^k. In this thesis, privacy of data related to temporal and spatial features is considered, use temporal hierarchy (i.e., a form of bottom-up aggregation)

to protect temporal information, and use generalization of location of interest to protect spatial features without losing information.

2.5.2 Privacy Preserving Perturbation Models

On the other hand, models that use perturbation method for preserving privacy in data publishing task include differential privacy and generative adversarial network (GAN) models:

- *Differential privacy* [CYXX17, JSB+13, ABCP13, XX15] is an alternative method for privacy preserving data publishing and has privacy guaranteed by using additive or multiplicative noise. Its main idea is addition of random noise in well-controlled data such that the new data are insignificantly different from the original dataset. Two types of random noise can be identified from research literature: additive and multiplicative. *Additive noise* adds a random value to each data set as described in Eq. (2.1):

$$N = D + \alpha \tag{2.1}$$

  where N is the publicly released noisy data, D is the original data, and α is the noise distribution. Thus, the original data can be reconstructed by a trusted party if the value of α is provided by reversing Eq. (2.1) [HZK13]:

$$D = N - \alpha \tag{2.2}$$

  Reversing back to the original data may not be precise depending on the size of D and the variance of the noise distribution α. Common additive noise distribution includes Gaussian distribution [HZK13] and Laplace distribution [GSW+17].

  Kim and Winkler [KW03] argued that *multiplicative noise* is more appropriate for modeling economic income data and discussed two approaches for constructing multiplicative noise. The first approach generates a random number with a mean $\mu = 1$ and a small variance before multiplying the original data D with the resulting value. The second approach involves several steps:

1. Logarithmic transformation of D, $D*$
2. Generation of covariance matrix of $D*$
3. Generation of multiplicative normal distribution random number with a mean $\mu = 0$ and variance/covariance multiplied by c (where $0 < c < 1$)
4. Addition of noise to $D*$
5. Computation of the anti-log of noisy data

Between the two types of random noise, additive noise parameter δ was explored by Kasiviswanathan and Smith [KS14] who defined differential privacy using Bayesian formulation and used probability distribution distance measure. The outcome is a relaxed notion of privacy having additive error parameter δ, which largely depends on the size of the dataset. Additive noise distribution depends on the values and distribution of the original dataset, whereas multiplicative noise has the advantage of ensuring uniform data protection using the noise coefficient of variation. Considering the uniformity of protection and simplicity of implementation, Brackenbury et al. [BOL19] designed a twin uniform multiplicative noise differential privacy for the protection of smart meter data. However, noisy data created using additive or multiplicative approach can be seen as a form of perturbation method, which often result in loss of data utility.

Hence, this research, rather than using additive or multiplicative noise, relates data using *temporal and spatial associations* (i.e., individual data points are related in time and space). With or without the addition of noise e, temporal and spatial grouping limits the amount of inference an adversary can obtain from a dataset [OLC21b].

- *Generative adversarial network (GAN)* [GPM+20] is a recent model for preserving privacy in big data emanating from deep learning research community. GAN is a deep neural network for generating sample data, computer vision [WSW21], images processing [WCY+20], speech recognition [DLP18] among other applications. GAN uses the theory of zero-sum game from the field of economics and

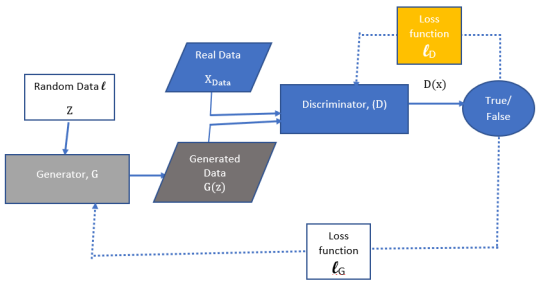


Figure 2.7: Generative adversarial network (GAN) architecture

game theory involving two players with the objective of achieving Nash equilibrium [Kre89] where the total gain and the total loss sum up to zero. Hence, GAN architecture (Figure 2.7) consists of two networks: (a) *generator G and (b) discriminator D* competing, training and learning from individual and collective weaknesses to optimize their performance towards achieving the balance of a zero-sum game. The objective of a generator network is to learn the distribution of original dataset to generate indistinguishable alternate dataset such that the discriminator is unable to differentiate the synthetic data from the real data, whereas the discriminator aims at differentiating between machine generated dataset from the real dataset in form of classification. The first step of GAN is similar to differential privacy, where a noise distribution is generated. However, the GAN generator is continuously trained to improve its performance such that the discriminator finds it difficult to differentiate the synthetic data from the real data. Both networks continuously learn from the training process, the generator uses back-propagation method [OC18] and minimizes the loss function to generate a new dataset that is very close but not identical to the actual original dataset, while the discriminator learns and updates its knowledge aiming at maximizing the likelihood of classifying real data as real and classifying machine-generated data as fake. Thus, for privacy protection, the Nash equilibrium can be seen as the privacy-utility balance zones as shown in Eq. (2.3). GAN network is a min-max problem, in which the generator, G is minimizing the loss function $\min_G$ while the discriminator, D is maximizing the

same loss function $\max_{D}$, and the objective is to achieve a zero-sum balance or Nash equilibrium:

$$\min_{G} \max_{D} \int(D, G) = \underset{x \sim pdata(x)}{E}[\log D(x)] + \underset{z \sim p_z(z)}{E}[\log(1 - D(G(z)))] \quad (2.3)$$

Similar to other neural network, GAN has its own challenges including interpretability and fairness of output, training convergence, collapse model issue [GPM+20], difficulty in attaining the Nash equilibrium due to the competing nature of the model. Moreover, all perturbation forms of privacy may distort the data beyond its usefulness. As a preview, this research work uses temporal and spatial correlation in grouping records to achieve two objectives namely:

- eliminate individual association with a record,
- maintain the temporal and spatial utilities of the dataset using hierarchical framework through bottom-up aggregation of temporal features and bottom-up grouping of location features in any dataset.

To recap, the summary of literature review of existing privacy preserving models is presented in Table 2.6.

Table 2.6: Privacy preserving model summary

Method	Model	Description	Drawback
Generalization and suppression	k-anonymity [SS98]	Equivalence class of K-indistinguishable records	Does not account for sensitive identifiers
	l-diversity [MKGV07]	Improves k-anonymity model to include at least l-*well represented* sensitive identifier in equivalence class	• Limitation of assumption of adversary knowledge [LLV07] • Distribution of the sensitive identifier may be skewed
	t-closeness [LLV07]	Improves l-diversity by ensuring that the distribution of the sensitive identifiers in the equivalence class is close to the distribution of the sensitive identifiers in the original dataset	Correlation between quasi and sensitive identifier may be lost as privacy increases
Perturbation	Differential privacy	Randomized noise addition such that the new data are insignificantly different from the original dataset	• Randomized noise control is subjective depending on dataset • Perturbation may distort the data beyond its usefulness
	Generative adversarial network (GAN)	Training two neural network - Generator and Discriminator with the objective of achieving a zero-sum balance or Nash equilibrium	• Difficulty in achieving Nash equilibrium • Perturbation may distort the data beyond its usefulness

2.6 Summary

The goal for the thesis is 3-fold. First, the design of a conceptual model for capturing and storing different variety of big data including structured, semi-structured and unstructured data along with metadata describing the data in support of open data FAIR principle of Findable, Accessibility, Interoperability and Re-usability for managing the data about data beyond the data life cycle. Second, design of privacy preserving model aiming at achieving balance between privacy and transparency concurrently with privacy and utility. Third, the application of the models to big data management tasks including data mining, data publishing, data visualization and data integration.

To achieve the goals of the thesis, this chapter extensively reviewed and discussed the following :

1. Big data type variety including structured, semi-structured and unstructured

data type.

2. Evolution of database models including hierarchical models, network model, relational models, objected oriented models, semi-structure models, data warehouse and graph models. The thorough review has shown that almost all database models have used graph theory in one form or the other. The hierarchical and network models are special cases of graphical representation, the relational models adopt set and relation theories; object-oriented models assume objects have states and behavior, semantic database models are based on knowledge representation and graph theories, Entity-relationship (ER) model combines semantic, set and relation, semi-structure models are tree-like models which are special case of graph where there is parent-child relationship enabling only top-down tree traversal. The review has also shown that schema design requirements and complex relationship modeling are the common challenges for database models. Graph database models evolved as part of NoSQL models enabling schema-less modeling of complex many-to-many relationship.

3. Big data privacy preserving models including k-anonymity, l-diversity and t-closeness designed for relational database combining the concept of generalization and suppression; differential privacy involving addition of random noise to the original dataset and generative adversarial network (GAN) model which is based on the concept of zero-sum game from the field of economics and game theory. The review has shown that all perturbation forms of privacy such as differential privacy and generative adversarial network may distort the data beyond its usefulness.

Chapter 3

Open Data Lake Architecture and Metadata Collection Framework

The key contributions of this chapter include the design and implementation of (a) an open data lake architecture (which is a non-trivial integration of existing open data initiatives and data lake architecture) and (b) a metadata collection framework (which is an extension of existing data collection framework). To elaborate, in this thesis, one of the goals is the design of an integrated conceptual architecture for collecting varieties of big data types including structured, semi-structured and unstructured raw data along with a data driven metadata collection framework design to enable and support open data FAIR principle of *findable, accessibility, interoperability and reusability*. The major contributions in this chapter are:

- Non-trivial integration of open data FAIR principle and data lake.
- Design of metadata collection framework for the open data lake

Open data lake refers to *an agile scalable repository for storing structured, semi-structured, and unstructured data from heterogeneous sources accessible to diverse community of users for different purposes and enabling data reusability using well defined metadata for data governance*. Key concepts for the open data lake are (a) diversity of data type (b) heterogeneity of data sources (d) accessibility to diverse community of users (e) metadata and governance supporting reusability of data. This

chapter presents a data-driven approach to designing data repository and the metadata for managing the ever-growing variety of big data types from heterogeneous sources.

3.1 Design of an Open Data Lake Architecture

The conceptual design for the open data lake is presented in Figure 3.1 comprising of (a) the data source, (b) the front-end user interface, and (c) the backend data repository with an integrated metadata for data governance.

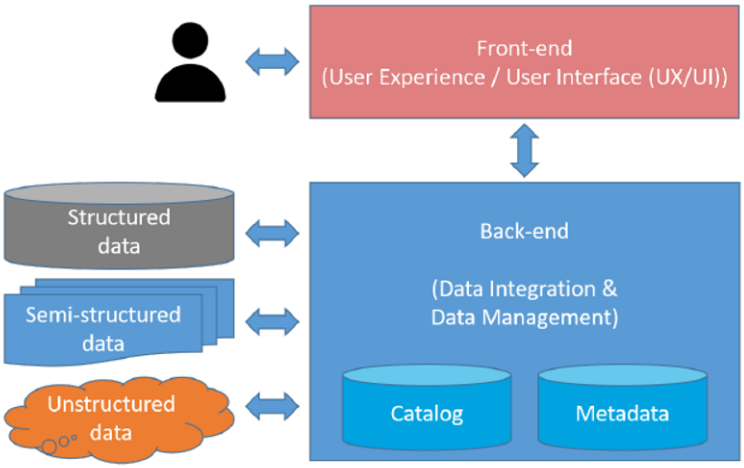


Figure 3.1: The proposed open data lake architecture

3.1.1 Data Sources

There are diverse heterogeneous data source to the lake, usually, data collection, origination or acquisition has source and purpose, for instance, statistics Canada collects data from various sources for further analysis to provide information such as population estimates, consumer Price Index (CPI), unemployment rates and other key economic indicators to the policy makers and the general public; the International Business Machine (IBM), The Weather Company (TWC) [1] continuously collects atmospheric data such as temperature, allergy tracker and air quality from diverse

[1] https://weather.com/

weather stations and satellites to provide weather forecast to various users of the weather app and researchers; arctic expedition data such as sub-bottom samples, salinity sample, bathymetric, side scan, sound velocity profile are collected for specific project with a well-defined objectives. Other data may be collected at random or for security purposes, for example, the prevalence of surveillance cameras in most households today are intended to provide monitoring of activities around individual properties, though the cameras are now helping the law enforcement agency in crime investigation. The data type can be structured, semi-structured or unstructured as discussed in Chapter 2. Structured data are typically arranged in rows and columns for easy access, aggregation, filtering and sorting. semi-structured data type has a delineating delimiters or tags to identify each element of the data. Examples include comma separated values (CSV), XML and JSON. Unstructured data type has no pre-defined data model and is characterized mostly by uncertainty and irregularity. Examples include portable document format (PDF), video, audio, pictures, surveillance cameras data, which are particularly complex containing combination of audio, video, image and pictures.

3.1.2 Front End

The **front-end user interface** of the open data lake architecture consists of the following three web interfaces:

1. The search interface, which allows users to find data and metadata available in the data repository using different keywords and filters.

2. The visualization (Viz) interface, which aims for visualizing subsets of any dataset that can be retrieved from the data repository.

3. The application programming interface (API), which allows developers and other third-party interactions. Users can interact with the data repository pragmatically to obtain a dataset rather than using the search or Viz interface.

3.1.3 Back End

The **back end** for the big data repository accepts structured, semi-structured, and unstructured data from heterogeneous sources and stored the data as-is. Each data source has rich metadata collected using a data driven approach and the data repository is surrounded by security layers to protect the ever-growing big data in the repository. On top of the data repository are three additional layers:

1. Access control layer, which controls the users access to the data repository.
2. Query processing layer, which accepts tasks from users through the search user interface and fetch the requested data to the user.
3. Visualization tool/plugin layer, which provides flexible options for users to visualize a subset of a search results requested through Viz in user interface.

3.2 Design of a Metadata Collection Framework

The design considers the nature of data collection and acquisition from the perspective of WHAT, WHO, WHERE, WHEN and HOW and translate the resulting output into a baseline metadata conceptual design, so as to comply with the "reusable" principle in the FAIR Principles

1. **WHAT:** Data collection and acquisition are done with specific goals. For instance, an expedition is usually project based. A project is initiated with clearly stated objectives and may have one or more events including equipment mobilization, testing and demobilization, sea trial, survey operation, data collection, and data analysis.
2. **WHO:** Who are the stakeholders? such as data owners or data collectors. In some cases, data collection and acquisition events are handled by multiple stakeholders. For instance, an expedition project team may include principal investigator, research collaborators, research assistant, marine surveyors, data scientists, technicians and local guards. Meanwhile, the data once collected is

useful for other users, for instance, the outcome of an expedition project will be useful for other stakeholders including researchers, policy makers and local communities.

3. **WHERE:** Data are collected from different locations (e.g., on-the-go through mobile devices, retail stores, social network platforms). For instance, for arctic expedition, events are undertaken at one or more sites within the Arctic region[2].

4. **WHEN:** The temporal metadata about a project include project initiation date, data collection period, data analysis, project closure and other project tasks period of execution. The temporal metadata may be simple, capturing only the timeline for the entire project or complex capturing individual task timelines, time lags and anticipated dependency of other projects over several years.

5. **HOW:** What are methodology and approaches for the project execution (e.g., data collection approach, method for data analysis and equipment deployment approach)? For instance, recall from Section 1.2, a collaborative prototyping methodology and long-term participatory design workshops were utilized to study physical and digital privacy concerns of older adults' adoption of technology. The how question also includes the funding sources.

The data-driven approach is applicable to any project enabling extraction of metadata at every stage of data life cycle management by storing the metal data separately. The metadata are available beyond the project life cycle and when the data are restricted for publication (which may be due to legal or privacy issues) or outdated.

Example 3.1. *Consider a survey operation project carried out between (WHEN) 07 October 2019 (as the start date) and 26 October 2019 (as the end date) inclusive. The objective* ***("what")*** *of the project named "Great Slave Lake Bathymetric Survey" is to collect bathymetric dataset for a proposed cable route in* ***("where")*** *the Great Slave Lake, Northwest Territory (NWT). The project team includes* ***("who")*** *the*

[2]https://www.state.gov/key-topics-office-of-ocean-and-polar-affairs/arctic/

principal investigator, a party Chief, two marine surveyors and an electronics technician. The project events include (a) equipment mobilization; (b) data collections along three identified submarine cable routes covering 133 km, 62 km and 372 km; (c) data cleaning; and (d) data analysis. The collection of multi-beam bathymetry dataset was carried out using ***("how")*** *a research vessel and a smaller launch vessel (e.g., rigid-hull inflatable boat (RHIB)) for survey and navigation support.*

3.2.1 Baseline Metadata Conceptual Design

Here, the baseline metadata are presented. The purpose of the metadata is to provide rich information about a data. The metadata of a data will always be available even when the data are restricted for publication due to legal or privacy issue and when the data are outdated. Metadata can be extracted at every stage of data life cycle management as shown in Figure 3.2. Big data life cycle management is made up of one or more ***tasks*** including:

- Data collection planning defining the purpose and use of the data.
- Data collection.
- Data quality control, data cleaning, and data integration.
- Knowledge discovery through Machine learning and data analysis.
- Data use, data sharing and reuse.

Data collection has both temporal and spatial components, data are collected from heterogeneous location ***site*** and by multiple ***stakeholders***. Data custodian has contact information. ***Data*** types and formats depend on purpose of collection, for instance, data collection during any Arctic or ocean expedition typically include hydrographic and bathymetric data, animal stock assessments, soil and salinity samples, snow samples, wind speed, air and water temperature. Data are collected during ***events/tasks*** using one or more ***equipment resources*** (e.g., research vessels, computer system, software, echo sounder, navigator, GPS receiver). The outcome of data

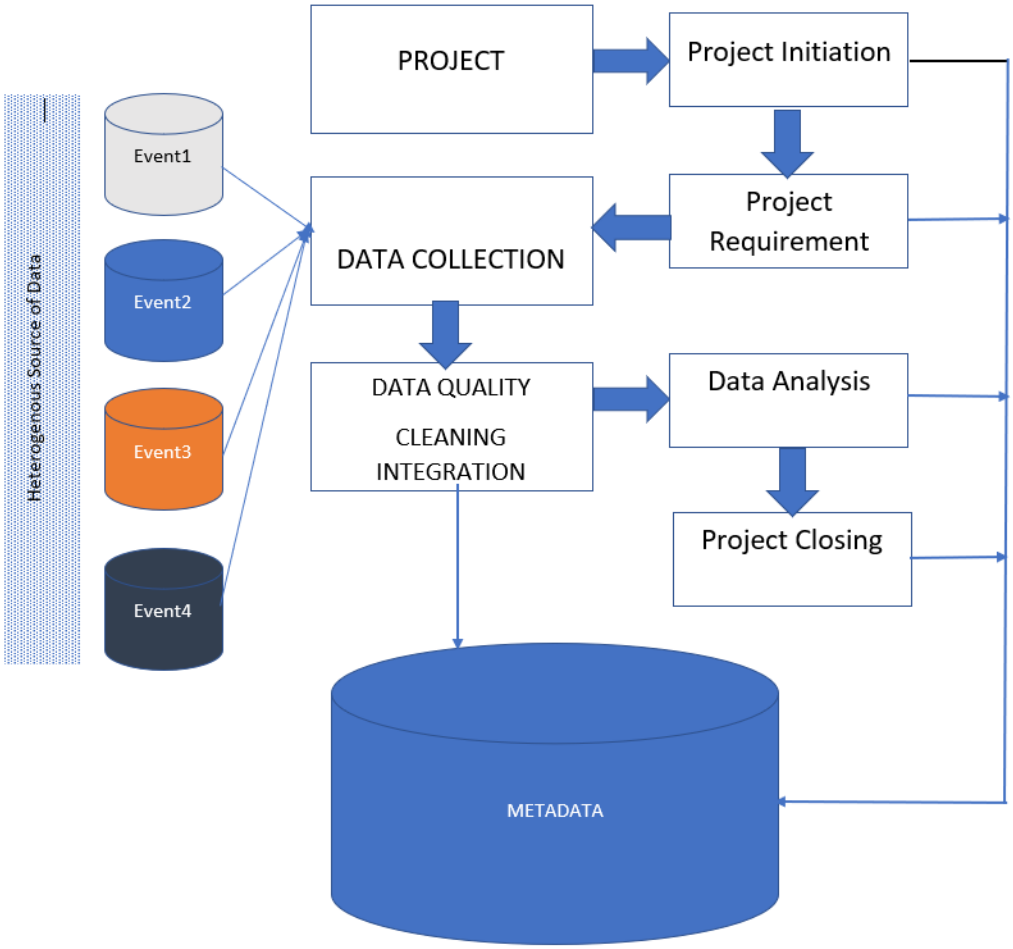


Figure 3.2: Project metadata collection flow

analysis may be shared with the public through publications, which are associated with ***publication*** citation count. Moreover, ***keywords*** when associated with the data collection event, purpose or use type ensure that the data are ***findable***, so as to comply with the "findable" principle in the FAIR Principles. Data license is also an important metadata to collect because it gives the users information on any constraints on re-usability of the data, when and how to contact the data owner and if payment is required to reuse the data if publicly available. Common license includes government open license [3] and Creative Common (CC) licenses [4] or government open

[3] http://open.canada.ca/en/open-government-licence-canada
[4] https://creativecommons.org/licenses

Table 3.1: Baseline metadata design

Field	**Description**
Project title	The title of the project such as Research Expedition of Great Slave Lake
Project DOI	Project unique identifier
Project description	Brief description of the project
Project purpose	The objectives of the project
Study site	The name of the study areas for example Arctic expedition are conducted with the Arctic region using NRCan standard such as Nunavut
Project status	The status of the project which may be in-progress or completed
Project start date	The project start date
Project end date	The project End date
Study site spatial bound	The north, south, east and west longitude and latitude of the study site
Project team and affiliation	The names of stakeholders, their institution affiliation and their roles including the Open Researcher and Contributor ID (ORCID) if applicable. Project team includes principal investigator, researchers and other collaborators;
Data license name	The usage license granted for the data. This may be creative common license or government open license.
Research equipment	List of equipment deployed for the project.
Project publication	The Title of publication related to the project if applicable.
Publication abstract	The summary of the publication including the methodology and results if applicable.
Publication citation	The citation for the publication if applicable.
Dataset name and description	The name and description of each data associated with the project.
Dataset citation	The citation for the dataset publication if applicable.
Project contact	The contact information of the principal investigator or the data owner.
Data access type	The access type may be public if there is no restriction and private if there is any legal or privacy issue restricting access to the data.
Keywords	Common tags associated with the data and the project including the project sites, subject area of the study, sample name and standardized dataset name such as bathymetry data.

licence.

The ***baseline metadata*** are presented in Table 3.1. This solution provides several benefits including: (a) The solution accepts all forms of data type, structured, semi-structured and to unstructured. (b) The metadata layer provides extra benefits. First, it supports machine learning to discover relationship between data and project as well as among projects. Second, it aligns the data repository with the FAIR (Findable, Accessible, Interoperability, and Reusable) principles of open data.

3.3 Evaluation

3.3.1 Comparison of the Open Data Lake Architecture with Existing Open Data Model and Data Lake

This section presents the evaluation of the open data lake conceptual model by comparing it with existing recent architecture namely: open data, data lake, data mesh and data fabric in Table 3.2.

Table 3.2: Open data lake, data lake, open data, data mesh and data fabric architecture

Metrics	Description	**Open data Lake**	Data Lake	Open Data	Data Mesh	Data Fabric
Data management principles	theoretical foundation	**Findable, Accessible, Interoperable, reusable (FAIR)**	None	FAIR	Discoverable, Addressable,Trustworthy, Self-describing, Interoperable, Secure (DATSIS)	None
Discoverability	Ease of data findings	**Centralized with data catalog, metadata and persistent object identifier**	None	Centralized	Decentralized	centralized with data catalog
Accessibility	Ease of data access Permission and security	**Centrally controlled front end access layer**	No access layer	Read only Access to the public	Decentralized domain-oriented.	centralized access control
Data governance	standard, policies, security and compliance management	**Centralized governance structure**	Centralized zone architecture	Centralized governance structure	Decentralised domain-oriented structure	Centralized governance structure
Data integration	Ease of data transformation	**Centralized data integration process**	Centralized data integration process with multi-layer of data preprocessing for pond architecture	Decentralized	Decentralized data integration process	Centralized data integration process
Latency	Time taken to deliver data to users	**Centralized architecture for low latency**	Centralized architecture		Decentralized architecture for high latency within domain	centralized architecture
Metadata framework	Metadata collection framework	**What, who, where, when, how (4WH)**	No framework			
Variety	Data Types	**Structured, semi-structured, unstructured**		Structured		
Scalability	Ability to handle increase in data volume	**Scalable centralized architecture**			Scalable decentralized architecture	Scalable centralized architecture

3.3.2 Implementation and Application of the Open Data Lake Architecture

This section presents a real-world application of the conceptual open data lake to real life big data collected for the Arctic research, which siloed in several repository including file system and databases.

In terms of **development environment and tools**, combination of the following tools was utilized:

- Amazon Web Services (AWS) Simple Storage Service (S3), which provides scalable object storage infrastructure through a web service interface for storing variety of data types.
- Amazon DynamoDB, which is NoSQL (i.e., "not only SQL") database service that supports key-value and document data structures for the catalog and metadata storage.
- Amazon Cognito: A scalable user access management infrastructure

In terms of real-life **datasets**, the evaluation of the design was carried out using, disparate data collected from heterogeneous arctic expeditions having diverse data types and format including:

- Portable Document format (PDF).
- *.txt for text data.
- *.PNG, *.JPEG for image data.
- Extended Triton Format (*.XTF), which is the industrial standard format developed by Triton Imaging inc. for record the raw backscatter and sidescan data.
- SEG-Y (*.segy), which is an open standard developed by the Society of Exploration Geophysicists (SEG) for storing geophysical data.

- Database file (*.db), which are QPS QINSy proprietary database files. Note that the QPS QINSy is a maritime software solution for survey planning and real-time hydrographic data processing.
- *.GDBTABLE, which is a geo-database table file created with ArcGIS software.
- *.HORIZON, which is a universal sandbox file containing the positions and velocities of objects.
- *.ATX, which is a geo-attributes index file created with ArcGIS software.
- *.XlS, which are Excel file data from mobile science labs, and renewable-energy powered plant production pods from Solar panels, wind turbines and diesel backup generator.
- Other semi-structured and unstructured data type such as audio, video, graph, charts, comma separated values (CSV), extensible markup language (XML), JavaScript object notation (JSON), Hypertext markup language (HTML) and links to other web pages.

In terms of **implementation details** of the open data lake architecture, they can be described as follows:

- Back-end data: The diverse big data from heterogeneous sources were stored using Simple Storage Service (S3). The S3 infrastructure inherently provide scalable data storage as object with or without modification. It is a bucket of objects such that we are able to centrally stored each project data individually as object within the central S3 bucket as presented in Figure 3.3. Each project has a persistent data object identifier (DOI) to link every dataset related to the project and its metadata. Each bucket has its security policy defined by the administrator.
- Metadata: THe metadata for each project was extracted and stored in DynamoDB, which is a *NoSQL* pair-value storage database within the same web services and create and index on the project DOI for easy information retrieval

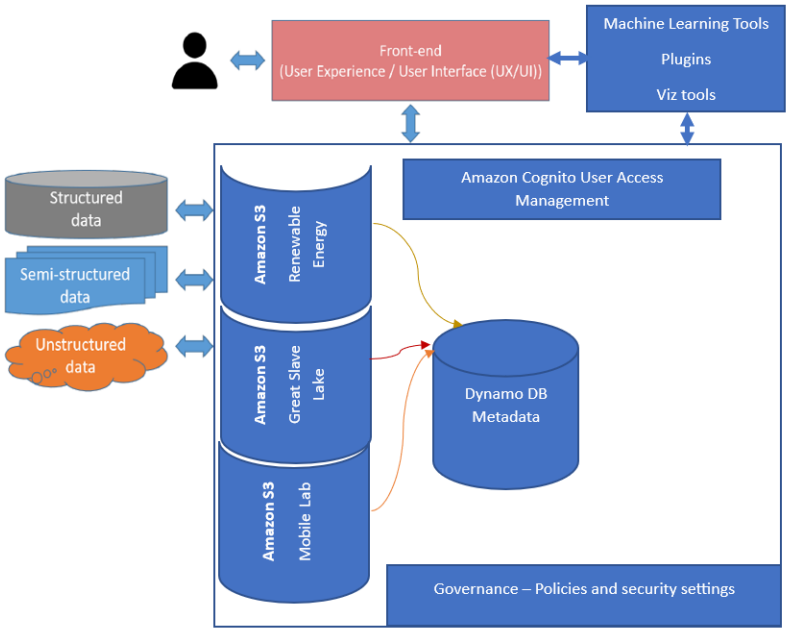


Figure 3.3: ARF open data lake architecture

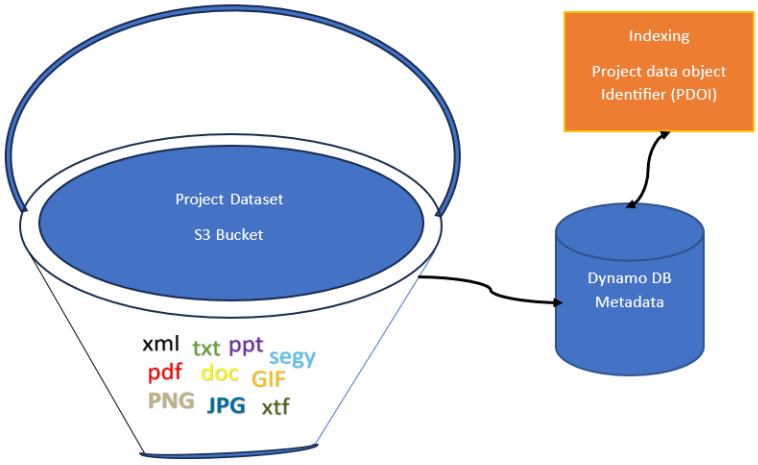


Figure 3.4: Project bucket

as depicted in Figure 3.4. DynamoDB provides support for CRUD operations (Create, Read, Update and Delete), triggers, auditing and data replication.

- User authentication: Amazon Cognito was utilized for users authentication. Amazon Cognito is an identity management infrastructure within Amazon web services supporting user directory, authorization and authentication across multiple applications. Users, majorly contributors are auto added to the user pool when they register on the user interface. The administrator is configured to have user and API authentication privileges including creating new user, adding user to a user group, auto confirm user registration, enable and disabling user within user group, deleting user pool and setting authentication policies.

- User access management: User access to each bucket within the S3 infrastructure was configured using Amazon Identity Access Management. The following roles were identified and configured:

  - Administrator, who has the highest level of authorization for performing all database operations including CRUD, user access management, triggers, auditing and replication management. The administrator is also responsible for configuration bucket resource policies.

  - Moderator, who is responsible for ensuring integrity of data upload into the lake. The moderator has *create* and *read* access to buckets.

  - Contributor, who is a member of the public submitting data for publication. The contributor has *upload* and *read* access to the bucket but has no back-end access to update the open data lake.

  - Developer, who is responsible for designing the user interface for the data lake and has *read/write* and *data encryption* access to the bucket in test environment.

- Data governance policies: Amazon S3 provides support for setting the retention and public availability policies for each dataset. The following data governance policies were considered:

  - Private: All data that are not available to the public

  - Public: All data available to the public.

- Legal hold date: Data with legal prohibition date. The data are only available after the hold date.
- Retain date: The retention date after which the data are not available to the public.

- Data encryption: AWS S3 supports data encryption in transit and at rest. There are three options for server-side encryption of data at rest namely, AWS S3 managed keys (SSE-S3), AWS key management services (SSE-KMS) and customer provided key (SSE-C). The default encryption policy was configured as SSE-S3 (Figure 3.5) which is the base level of encryption allowing auto encryption of data on upload by the contributors.

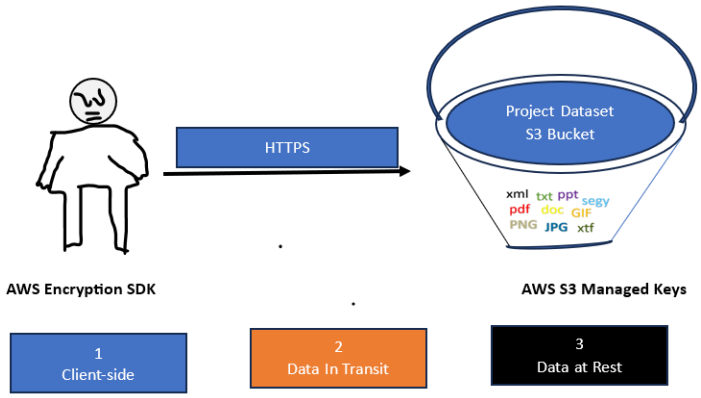


Figure 3.5: Data encryption process in AWS

- Data Integrity Check: We configured multi-part uploads checksum SHA-256 algorithm using the S3 API allowing splitting and concatenation of large files by S3 during the upload process by the contributors.

3.3.3 Application of the Metadata Collection Framework for Arctic Research

We applied the 4WH metadata collection framework for the arctic expedition big data by collecting data relating to "what" as shown in Figure 3.6, "who" as shown in Figure 3.7, "where" as shown in Figure 3.8 and "how" as shown in Figure 3.9.

WHAT?

Project Description
Title
Objectives
DOI
Publication Citation
Dataset Description
Data License
Data Access Type
Keywords

Figure 3.6: "What?"

WHO?

Project Contributors		
Name	Role	Contact information
*	Principal Investigator	
	Choose an item.	
	Choose an item.	
	Choose an item.	
	Choose an item.	

Project Contact	
Name	
Affiliation	
Email	
Phone	
Address	

Other Stakeholders	
Name, Affiliation, Email	
Name, Affiliation, Email	
Name, Affiliation, Email	
Name, Affiliation, Email	
Name, Affiliation, Email	

Figure 3.7: "Who?"

WHERE?

Location 1	Spatial bound
Location 2	Spatial bound
Location 3	Spatial bound
Location 4	Spatial bound
Location 5	Spatial bound

Figure 3.8: "Where?"

HOW?

Research Equipment		
Grant Information		
Award Title		
Funder Name		
Grant Number		
Methodology		

Figure 3.9: "How?"

3.4 Summary

In this chapter, (a) an open data lake architecture and (b) a metadata collection framework were designed and implemented . The former can be considered as a non-trivial integration of existing open data initiatives and data lake architecture, whereas the latter can be considered as an extension of existing data collection framework. To evaluate the design and implementation, Key features of the open data lake architecture were compared with those of the existing open data model, data lake, data mesh and data fabric. The chapter also showcased a real-world application of the metadata collection framework with a case study for Arctic research expedition. Note that, although the metadata collection framework was designed and implemented for Arctic research, it can be easily generalized for many other real-world application domains.

To elaborate, in this chapter, the conceptual model for storing variety of data types including structure, semi-structured and unstructured data types combining the architecture framework of data lake, the FAIR principle of open data concept was presented. A three-layers architecture was proposed having data source, the user interface and the back-end layers. The data source layer accepts varieties of data types including structured, semi-structure and unstructured data types. The user interface layer is further divided into three namely search interface enabling searching and filtering by keywords, visualization interface enabling actionable and insightful visualization of data subset and finally, the application programming interface (API) enabling developer and other third parties interaction with the open data lake. The third layer is the Back-end comprising of data governance structure, access and permission control, query processing and integrated visualization tools. Secondly, a framework for metadata collection was designed to ensure interoperability of metadata across and beyond organization use of the data and to ensure reusability of the data beyond its life cycle. The metadata framework enables heterogeneous big data integration and machine learning processing and pattern discovery. Thirdly, the open data lake model was compared with data lake, data mesh and data fabrics. While all the four architectures have common goals of solving big data management variety, volume and accessibility issues, the open data lake provides a framework for metadata collection to enable machine learning pattern discovery beyond the data useful life cycle and enable data integration with ease. Lastly, the conceptual model was applied to big data management of variety of data types collected from heterogeneous sources in Canada arctic region using Amazon web services cloud infrastructures comprising of simple storage services (S3) bucket, Amazon Cognito and dynamo $NoSQL$ pair-value storage database for implementation. The next chapter will discuss the details of the big data hierarchical spatio-temporal model aspect of this thesis.

Chapter 4

Big Data Hierarchical Spatio-temporal Model (HSTM)

After designing and implementing an open data lake architecture with a metadata collection framework, let us deal with spatial and/or temporal data in this architecture. The key contributions of this chapter include the design and implementation of (a) a hierarchical spatio-temporal model (HSTM) (which integrates and links a hierarchical temporal model with a hierarchical spatial model) and (b) a spatial representative point (which represents spatial locations of data points in temporal sequences).

To elaborate, this chapter presents the overview of the proposed hierarchical spatio-temporal models for big data management tasks as shown in Figure 4.1. The chapter describes the three components of the solution. Specifically, the key concepts behind the HSTM—namely, (a) temporal hierarchy (with extension to time generalization for low level granularity), (b) location hierarchy and (c) temporal representative points. With temporal hierarchy, the HSTM support big data analytics on temporal granularity, the location hierarchy supports various level of granularity analytics on different hierarchies. Moreover, temporal representative points enable the HSTM to represent a collection of points within a specific level of temporal and/or location hierarchy. Furthermore, the HSTM adapts the temporal representative points from temporal data to spatial data, and it computes representative points over spatial

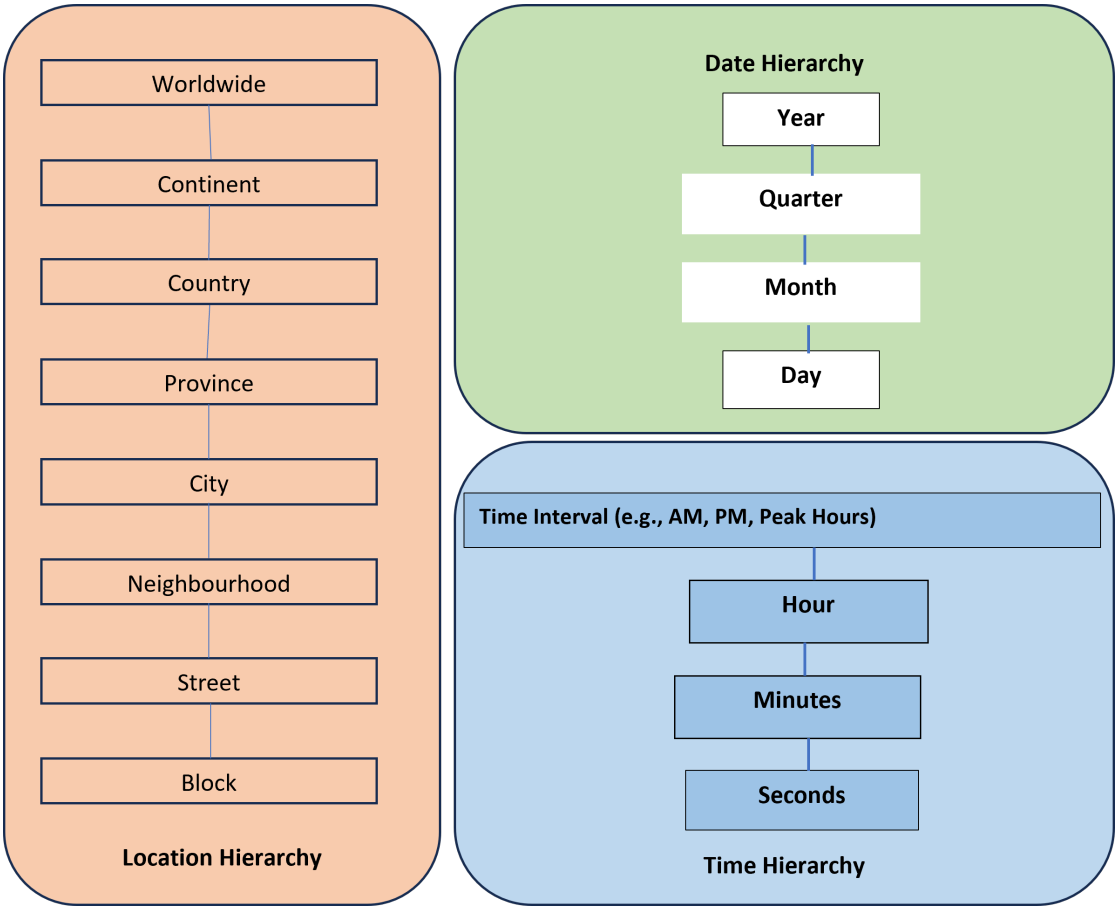


Figure 4.1: An overview of the proposed solution

data to generalize spatial locations.

4.1 A Temporal Hierarchical Model

Temporal hierarchy is defined as a non-overlapping temporal aggregated frequency designed over predefined periodic levels [OLC20b]. It is similar to a bottom-up generalization, but with a condition that the parent is exactly an aggregate of its descendants. To elaborate, the smallest unit of time (e.g., hour) is at the leaf, and the aggregation is constructed bottom-up to the root. Each temporal hierarchy level has business implication. For example, operational level management may be interested in detail transaction by the hour for daily decision, staff scheduling, or resource allo-

cation. Monthly, quarterly and annual aggregations are useful for both tactical and strategic levels of management for budgeting and several other business decisions.

Hence, a 4-level temporal hierarchy is designed with (a) days at the bottom, (b) months and (c) quarters at intermediate levels, and (d) the year at the top (i.e., root) of the hierarchy. See Figure 4.1. Generally, each level is an aggregate of its intermediate level below. For example, the year is an aggregate of 4 quarters:

$$Y = \sum_{i=1}^{4} Q_i \tag{4.1}$$

Each of the four quarters is an aggregate of the corresponding 3 months. For example, Quarter 1 is an aggregate of the first 3 months in a year (i.e., January, February and March):

$$Q_i = \sum_{j=1}^{3} M_j \tag{4.2}$$

Similarly, each of three months within a quarter is an aggregate of the corresponding number of days (e.g., ranging from 28 to 31) within the month. For example, Month 1 (i.e., January) is an aggregate of its 31 days:

$$M_j = \sum_{k=1}^{d} D_k \tag{4.3}$$

where d is the maximum number of days within the Month j. For example, $d = 31$ for January. Finally, each day is an aggregate of its 24 h:

$$D_n = \sum_{n=1}^{24} H_n \tag{4.4}$$

The temporal hierarchy from day up to year (i.e., *date generalization*) works well for preserving privacy of health-related real-life applications. However, for non-health related applications (e.g., parking), details of data go to a much finer granularity (e.g., with timestamp). Thus, the temporal hierarchy is extended to *time generalization* for each day.

Figure 4.2 shows the complete temporal hierarchy for date and time generalization, in which daily information D_k can be obtained by aggregating information from both

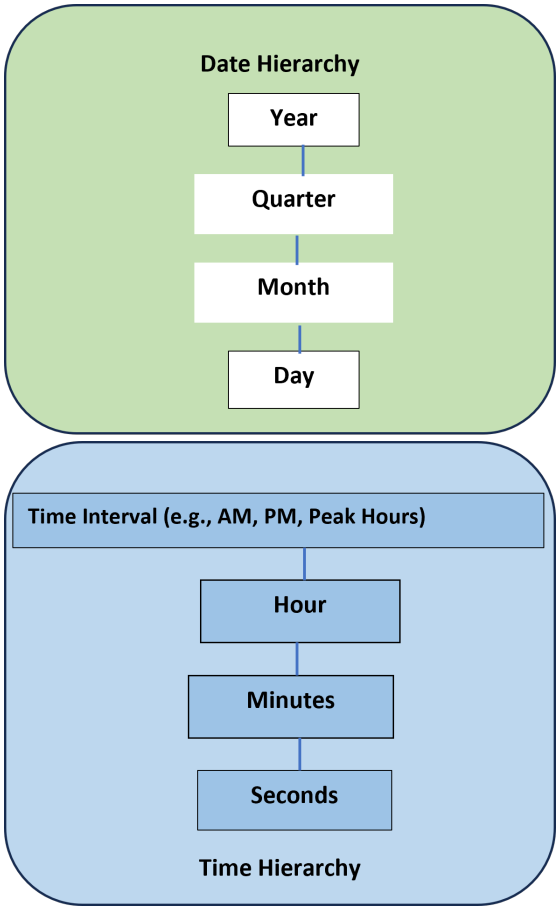


Figure 4.2: Temporal hierarchy

daytime DT and nighttime NT, where (a) DT is obtained by aggregating 12 hourly information HH_i from daytime (say, 06:00-18:00) and (b) NT is obtained by aggregating the remaining 12 hourly information HH_i from nighttime (say, 18:00-06:00). Along this direction in the temporal hierarchy extension (to time generalization), hourly information H_i is obtained by aggregating information from its 60 minutes MM_j, each of which is obtained by aggregating information from its 60 seconds SS_k.

Mathematically, this extended temporal hierarchy can be represented as:

$$D_k = DT + NT \tag{4.5}$$

$$DT \text{ (and } NT) = \sum_{i=1}^{12} HH_i \tag{4.6}$$

$$HH_i = \sum_{j=1}^{60} MM_j \tag{4.7}$$

$$MM_j = \sum_{k=1}^{60} SS_k \tag{4.8}$$

The detailed time of events can be generalized to the next level in the hierarchy (e.g., to preserve the privacy of individual record within the dataset).

4.2 A Spatial (Location) Hierarchical Model

Spatial hierarchy helps represent spatial geometric data at the appropriate level of hierarchy in the sense of revealing essential spatial geographic location information for data analytic, data visualization, data mining or data prediction. Similar to the aforementioned temporal hierarchy, spatial data in the spatial hierarchy can also be aggregated. For example, information in a several related blocks (e.g., frequency counts of events happened in these blocks) on the same street can be aggregated. The aggregated information from several streets within the same neighborhoods can be collected and grouped. Along this direction, aggregated information from multiple neighborhoods can be further aggregated to form information for a city. Information from several cities in a province (or state) can then be grouped to form aggregated data for a country. Information for different countries within a continent can be aggregated, and the resulting information can be further aggregated to form the global data. The spatial hierarchy model for location aggregation is presented in Figure 4.3.

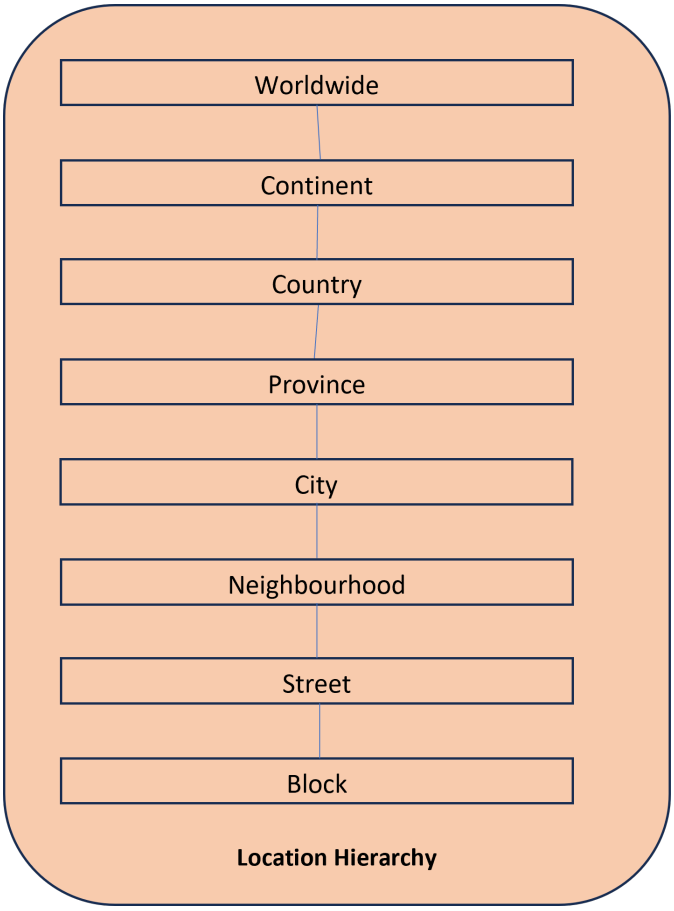


Figure 4.3: Location hierarchy

4.3 Data Linkage and Integration to Form a Hierarchical Spatio-temporal Model

Given that temporal data for the temporal hierarchical model and spatial data from the spatial hierarchical model may be coming from the same (i.e., homogeneous) rich data sources or different (i.e., heterogeneous) rich data sources, this section explores how to integrate these data and link them for a hierarchical spatio-temporal model (HSTM). *Key ideas* behind the data linkage and integration can be described as follow. First, identify temporal and spatial (location) features with the dataset. Then, identify category of event by adding category of events to the grouping factor. Finally, use the temporal hierarchy from each dataset for the linkage to form a HSTM.

Algorithm 4.1 Hierarchical temporal data integrator

```
Input: Spatio-temporal dataset from heterogeneous sources D
Output: Spatio-temporal aggregated dataset D'
for each dataset D do
    Date ← create temporal hierarchy
    for selected temporal hierarchy do
        for each item in temporal hierarchy in D do
            group by Location, Event Category
            compute frequency count
            compute Representative point for x, y coordinates
Aggregate Temporal hierarchy t_i, location hierarchy l_i, event category c_i and
frequency count f_i
```

A challenge in the integration task was the identification of category for each dataset. To deal with the challenge, feature selection was done by mapping category attribute for each dataset. Then, common attributes were selected for each dataset for integration.

Moreover, data view option was explored since each of the dataset can be tabulated by the following steps:

1. Create a dimension for each of the dataset.
2. Create temporal and location indices.
3. Create a view.

The aforementioned spatio-temporal data linkage/integrator model can be summarized in Algorithm 4.1 and Figure 4.4.

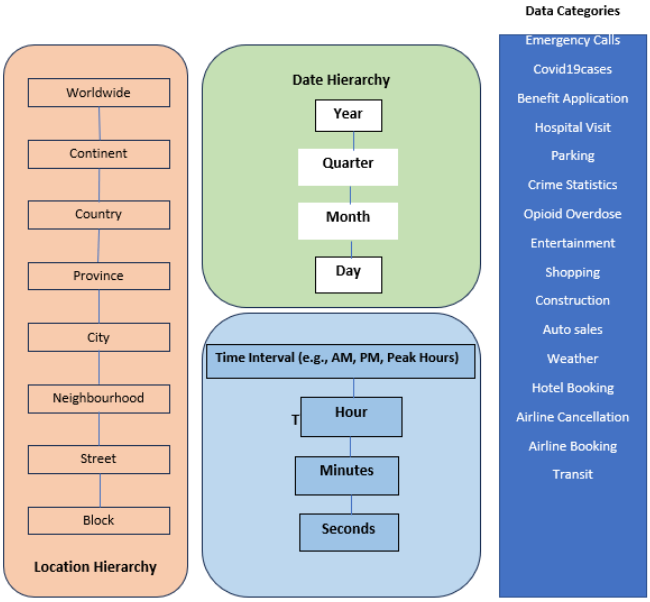


Figure 4.4: Temporal hierarchy big data integration

4.4 Spatial Representative Point for the Hierarchical Spatio-temporal Model

This section investigates different approaches to representing group of related *spatial data* for the spatio-temporal model and discuss merit and demerit of each approach.

First, *a spatio-temporal representative point* $R(x, y)$ is defined as a point representing all n related points $(x_i, y_i)_{i=1}^{n}$ in a line, multi-point, a polygon or a group of observations within a specific level of spatio-temporal hierarchy [OLC20a].

A common approach in research literature [AC14] to represent group of spatial data is clustering based on distance and proximity of spatial point to each other; an obvious issue with this approach is that outliers are difficult to identify which may result in outcome bias. Then, another option is "can we pick a point at random"; obviously, that is possible and there is an assurance that the point is within the group of related observations, however, random choice has inherent issue of inconsistency, that is, different random point is generated at run time each time and secondly, the random choice may be an outlier that may too close to the bounding edges of the

points. Then, another consideration is the center of gravity of the observation which is also known as centroid. Eventually, combining the random and centroid to form a hybrid representative point. Each approach is discussed as follows:

- A ***random point***, which is randomly selected from all n data points within a specific temporal hierarchy to represent a common spatial location within that temporal hierarchy:

$$R(x, y) \in (x_i, y_i)_{i=1}^{n} \tag{4.9}$$

  Although *a random point* is loosely computed random point that represents geographical points (or multi-point), it is guaranteed to be a real data point within the objects (e.g., geographical points) that it represents.

- ***Centroid***, which is an average (or mean center) of the n coordinates or a simple average of x and y coordinates:

$$R(x, y) = \left(\frac{\sum x_i}{n}, \frac{\sum y_i}{n} \right) \tag{4.10}$$

  In other words, a centroid is the center of gravity of the geographical points (or polygon or multi-point) regardless of whether it is an actual data point within the geographical points.

- ***Weighted representative point*** $R(x, y)$, which can be computed by using weighted mean centre of the n coordinates or the minimum square distance within existing points [1]. The minimum square distance $Dist_{min}$ is given as:

$$Dist_{min} = \min \sum \left[(x_j - x_i)^2 + (y_j - y_i)^2 \right] \tag{4.11}$$

- ***Minimum bounding box***, which is the rectangular bound for the set of point represented as shown in Eq. (4.12). Using this approach, the centroid of the bounding box is calculated.

$$\text{bounding box} = [\min(x), \min(y), \max(x), \max(y)] \tag{4.12}$$

[1] https://www150.statcan.gc.ca/n1/pub/92-195-x/2011001/other-autre/point/point-eng.htm

- ***Convex hull***, which is defined as the smallest convex object containing all points in a given set of points observations. Convex hull encloses all points within the smallest convex object. With this approach, the weighted average of the vertices of the convex hull with the assumption that a convex hull contains less points than the original dataset, there is possibility that convex hull approach may give lesser error and may as well be faster than finding the weighted average of the entire points population.

- ***Medroid***, which can be considered as an enhanced temporal representative point calculated by selecting the point closest to the center of gravity. A random temporal representative point is a real data point. So, it helps reveal characteristics of the group of observations it represents. Hence, it would be valuable for applications in which one would like to discover and understand knowledge on key characteristics of the group. however, random point are subjective because they are selected at run-time and may probably be an outlier. On the other hand, centroid is the center of all points. So, it helps reveal the "center" of the group of observations it represents. As it is not selected from real data points, it is more likely to preserve privacy better. Hence, it would be valuable for applications in which one would like to preserve more privacy. This is a trade-off between knowledge discovery and privacy preservation.

  As an enhancement, to avoid bias resulted from random selection (e.g., randomly selecting an edge or outlying point as a temporal representative point), several candidate temporal representative points were drawn and the one that is closest to the center of gravity was selected. By doing so, the benefits of both worlds were achieved. This enhanced random temporal representative point is a real data point that is near the "center" of the group of observations it represents and help reveal the characteristics of the group.

Example 4.1. *In Buffalo parking dataset, there are 52 tickets issued on 05 May 2019 on CLEVELAND AVE, block 2002 as shown in Figure 4.5, However, the convex hull has only 14 points as shown in Figure 4.6. Also, the bounding box resulted in four points, one at each corner of the box shown in Figure 4.7. In all, there are 5 different*

points that can be used to represent the spatio-temporal hierarchy of the ticket issued at Cleveland Ave, Block 2002, on 05 May 2019 as shown in Figure 4.8.

1. *The blue marker is the weighted representative point*
2. *The green marker is the convex hull centroid*
3. *The pink marker is the random choice representative point*
4. *The red marker is the medroid*
5. *The black marker is the bounding box centroid*

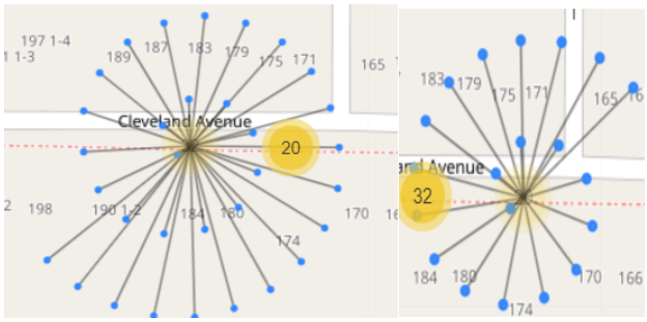


Figure 4.5: Parking tickets for Block 2002 on 05 May 2019

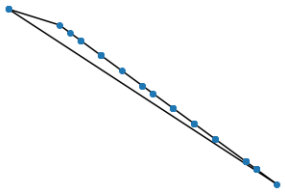


Figure 4.6: A convex hull for Block 2002

Note from Table 4.1 that the medroid and the random choice are actual points selected within the population of the given set of points. Thus, the two approaches can be traced back to the original summon number (ticket). The question arising from selecting either the random choice point or the medroid is whether we want to use an actual point that can be traced back to the ticket and subsequently to the

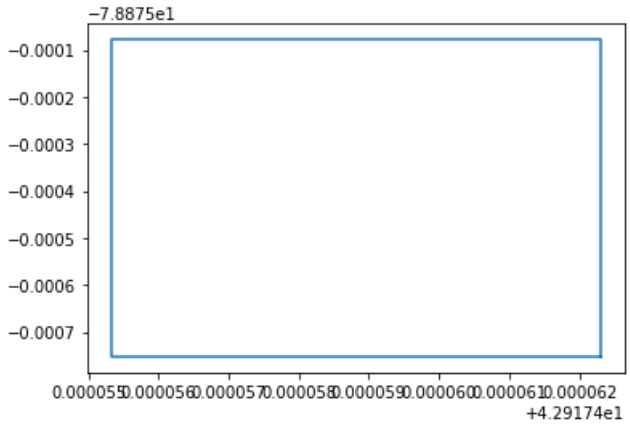


Figure 4.7: A bounding box for Block 2002

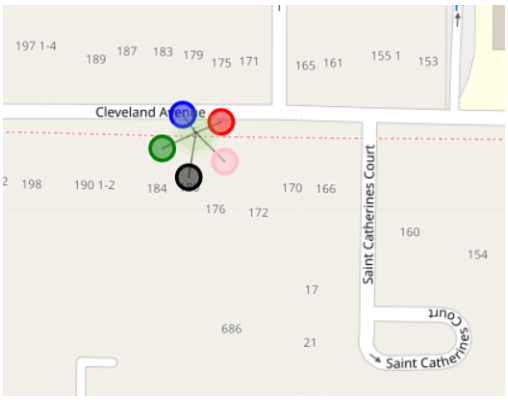


Figure 4.8: Five representative points for Block 2022: (a) weighted representative point in blue, (b) convex hull centroid in green, (c) random choice in pink, (d) medroid in red, and bounding box centroid in black.

Table 4.1: Representative point

Method	Summon number	Points
Centroid		(42.9174590040163, -78.8753998699768)
Medroid	R2269366	(42.9174590794293, -78.8754021461174)
Random	62269646	(42.9174614955121, -78.8753729774922)
Convex hull		(42.9174589451838, -78.8754074864182)
Bounding box		(42.9174588199990, -78.8754153799990)

owner or do we want to use calculated points- the centroid, convex hull or bounding box representative point.

Example 4.2. *See Tables 1.2, 1.3 and 4.2, respectively, for sample parking data, hospital visit data and emergency call data collected in space (Winnipeg) and time (12 March 2018). As they are related by spatio-temporal commonality, these three tables can be integrated and linked to form Table 4.3.*

Table 4.2: Winnipeg emergency call data

Incident Number	Incident Type	Call Time	Closed Time	Ward
2018027221	Medical Emergency	03/12/2018 04:29:59 PM	03/12/2018 06:57:48 PM	Charleswood Tuxedo Westwood
2018027223	Medical Emergency	03/12/2018 04:36:47 PM	03/12/2018 05:00:39 PM	Elmwood East Kildonan
2018027224	Medical Emergency	03/12/2018 04:40:42 PM	03/12/2018 06:23:15 PM	St. Boniface
2018027227	Medical Emergency	03/12/2018 04:47:16 PM	03/12/2018 06:52:06 PM	Elmwood East Kildonan
2018027228	Medical Emergency	03/12/2018 05:00:59 PM	03/12/2018 06:15:39 PM	Charleswood Tuxedo Westwood
2018027229	Medical Emergency	03/12/2018 05:00:42 PM	03/12/2018 06:35:04 PM	St. James
2018027234	Medical Emergency	03/12/2018 05:14:07 PM	03/12/2018 06:16:35 PM	Mynarski
2018027236	Medical Emergency	03/12/2018 05:15:12 PM	03/12/2018 05:20:01 PM	Mynarski
2018027241	Medical Emergency	03/12/2018 05:29:29 PM	03/12/2018 07:44:03 PM	Point Douglas
2018027243	Medical Emergency	03/12/2018 05:29:36 PM	03/12/2018 05:39:56 PM	Daniel McIntyre

Table 4.3: Winnipeg multi-event data integration

Date	Event Category	Frequency
March 12, 2018	Medical Emergency call	10
March 12, 2018	Parking	7
March 12, 2018	Hospital Admission	8

4.5 Evaluation

4.5.1 Experimental Evaluation on the HSTM

4.5.1.1 Experimental Setup

The data integration and linkage procedure was evaluated using several real-world data. Specifically, in terms of **datasets**, four datasets were integrated and linked namely: (a) emergency call dataset, (b) parking data, (c) Substance-use dispatch data and (d) naloxone administration data

Table 4.4: Winnipeg emergency call data description

Attribute	Description	Data type
Incident number	Incident identifier	text
Incident type	Type of emergency call	text
Call time	Date and time of emergency call	date
Closed time	Emergence resolution date and time	date-time
Motor vehicle incident	Boolean attribute for motor vehicle incident	Boolean
Unit	Responsible work unit for dispatch assignment	text
Ward	City ward	text
Neighborhood	City Neighborhood	text

Details of these real-life Canadian open datasets from City of Winnipeg and Province of Manitoba can be described below:

- Emergency call dataset[2], which is the emergency call data containing 733,078 emergency call responded to by the fire and rescue team of the City of Winnipeg. Details of attributes are shown in Table 4.4.

- Parking ticket dataset[3], which is the City of Winnipeg parking ticket transaction between January 2010 and September 2022. See Table 4.5.

- Substance-use dispatch dataset[4], which contains record of substance-use events within the City of Winnipeg. The substance record includes alcohol marijuana, cocaine, crystal meth and opioid from January 2011 to September 2022 containing 76,210 substance-use events. Details are shown in Table 4.6.

- Naloxone administration dataset[5], which is the data containing patient age group, gender, neighborhood ward of Naloxone administration in the City of Winnipeg from 2007 to 2022. Naloxone is the drug used for treating opioid overdose. The drug helps counter the toxic effect of opioid. The dataset description is presented in Table 4.7.

Summary of the four datasets is shown in Tables 4.8 and 4.9.

[2]https://data.winnipeg.ca/Fire-and-Rescue-Response/WFPS-Call-Logs/yg42-q284

[3]https://data.winnipeg.ca/Tickets-and-Adjudication/Parking-Contravention-Citations-/bhrt-29rb

[4]https://data.winnipeg.ca/Fire-and-Paramedic-Service/Substance-Use/6x82-bz5y

[5]https://data.winnipeg.ca/Fire-and-Paramedic-Service/Narcan-Administrations/qd6b-q49i

Table 4.5: Winnipeg parking ticket data description

Attribute	**Description**	**Data type**
Issue date	The issue date and time of the parking ticket	date
Violation	A description of the violation	text
Street number	The street number where the parking violation occurred	text
Street name	The street name where the parking violation occurred	text
Discounted fine	The discounted fine for the parking violation	float
Full fine	The original fine associated with the parking violation	float
Location	The geographic location (latitude, longitude) of the parking violation	text

Table 4.6: Winnipeg substance-use data description

Attribute	Description	Data type
Incident number	Incident identifier	text
Dispatch date	Event reported date	date
Patient number	Patient number	integer
Age	Patient age group	text
Gender	Patient gender	text
substance	Type of substance-use	text
Ward	City ward	text
Neighborhood ID	Neighborhood identification	text
Neighborhood	City neighborhood	text

Table 4.7: Winnipeg naloxone administration data description

Attribute	Description	Data type
Incident number	Incident identifier	text
Dispatch date	Date of drug administration	Date
Patient number	Patient number	integer
Age	Patient age group	text
Gender	Patient gender	text
Ward	City ward	text
Neighborhood ID	Neighborhood identification	text
Neighborhood	City neighborhood	text
Narcan administrations	Number of naloxone dose administered	integer

Table 4.8: Dataset summary

Dataset	Size (MB)	#rows
Emergency call	80.30	733,078
Parking Tickets	134.00	1,765,299
Substance-use	6.89	76,210
Naloxone administration	1.39	16,711

Table 4.9: Winnipeg open data temporal year integration

Year	NaloxoneAdmin	SubstanceUse	EmergencyCall	Parking
2007	4			
2008	265			
2009	526			
2010	711			167,418
2011	790	5,068		164,719
2012	873	4,270		164,137
2013	691	4,226		143,708
2014	846	4,786	13	151,691
2015	1,096	5,512	61,566	158,964
2016	2,321	6,747	74,018	147,734
2017	2,303	7,756	92,929	169,502
2018	1,588	7,774	101,584	144,792
2019	2,018	8,069	109,319	133,324
2020	3,789	7,631	104,361	71,618
2021	4,076	8,141	114,278	68,521
2022	2,859	6,230	75,010	79,161

4.5.1.2 Processing Time for Data Integration and Linkage at Different Hierarchical Levels

To integrate and link these four real-world open data, features selection was done by mapping category attribute for each dataset:

- Emergency dataset $\mapsto$ incident type
- Parking dataset $\mapsto$ violation
- Substance-use dataset $\mapsto$ substance
- Naloxone administration dataset $\mapsto$ preset to NaloxoneAdmin

Next, varied the selection of location attributes from neighborhood to city for each dataset:

- Emergency dataset: Neighborhood $\rightarrow$ city
- Parking dataset: As it has only street information and points, the default is set to city location level.
- Substance-use dataset: Neighborhood $\rightarrow$ city
- Naloxone administration dataset: Neighborhood $\rightarrow$ city

Then, common attributes were selected for each dataset for integration. The processing time recorded for the temporal hierarchy variation for year, month and week is presented in Table 4.10.

Table 4.10: Processing time (in seconds)

Location hierarchy	Day	Week	Month	Year
Neighborhood	236	238	219	219
City	228	228	212	212

Moreover, data view option was explored by creating dimension for each of the dataset, temporal and location indices, as well as a view by using the following query:

```
(SELECT COUNT(IncidentNumber) AS Count,
  IncidentType AS IncidentType,
  EmergencyCall AS Category,
  YEAR(DispatchDate) AS Year,
  MONTH(DispatchDate) AS Month,
  DATENAME(W,DispatchDate) AS Weekday,
  DATEPART(Q,DispatchDate) AS Quarter,
  Neighbourhood AS Location
 FROM dataIntegration.dbo.CallLog
 GROUP BY IncidentType,
  YEAR(DispatchDate),
  MONTH(DispatchDate),
  DATENAME(W,DispatchDate),
  DATEPART(Q,DispatchDate), Neighbourhood)
UNION
(SELECT COUNT(IncidentNumber) AS Count,
  Substance AS IncidentType,
  SubstanceUse AS Category,
  YEAR(DispatchDate) AS Year,
  MONTH(DispatchDate) AS Month,
  DATENAME(W,DispatchDate) AS Weekday,
  DATEPART(Q,DispatchDate) AS Quarter,
  Neighbourhood AS Location
 FROM dataIntegration.dbo.Substance
 GROUP BY Substance,
  YEAR(DispatchDate),
  MONTH(DispatchDate),
  DATENAME(W,DispatchDate),
  DATEPART(Q,DispatchDate), Neighbourhood)
UNION
(SELECT SUM(NarcanAdmin) AS Count,
  NarcanAdmin AS IncidentType,
  Narcan AS Category,
  YEAR(DispatchDate) AS Year,
  MONTH(DispatchDate) AS Month,
  DATENAME(W,DispatchDate) AS Weekday,
  DATEPART(Q,DispatchDate) AS Quarter,
  Neighbourhood AS Location
 FROM dataintegration.dbo.NarCan
 GROUP BY YEAR(DispatchDate),
  MONTH(DispatchDate),
  DATENAME(W,DispatchDate),
```

```
  DATEPART(Q,DispatchDate), Neighbourhood)
UNION
(SELECT COUNT(violation) AS Count,
  Violation AS IncidentType,
  Parking AS Category,
  YEAR(IssueDate) AS Year,
  MONTH(IssueDate) AS Month,
  DATENAME(W,IssueDate) AS Weekday,
  DATEPART(Q,Issuedate) AS Quarter,
  Street AS Location
 FROM dataintegration.dbo.ParkingWpg
 GROUP BY Violation, YEAR(IssueDate),
  MONTH(IssueDate),
  DATENAME(W,IssueDate),
  DATEPART(Q,Issuedate), Street)
```

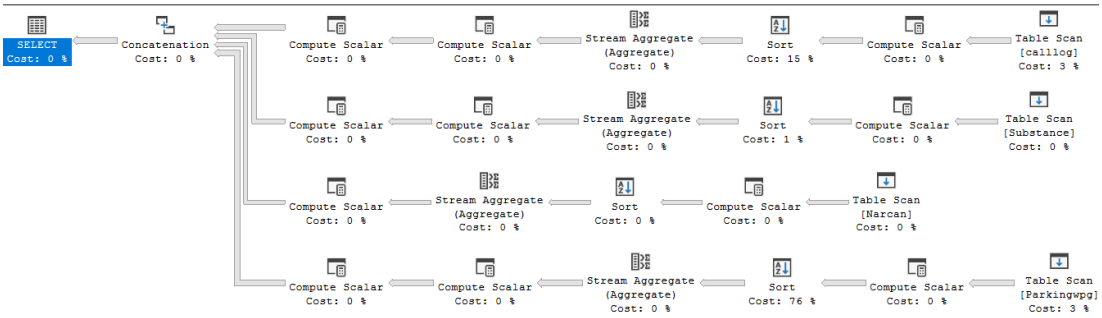


Figure 4.9: SQL query execution plan

The view query processing time was 10 seconds. The query execution plan shown in Figure 4.9 shows that 76% of the query execution time was spent on sorting the parking dataset, 15% on sorting the emergency call dataset, 3% on scanning parking dataset, another 3% for scanning emergency call dataset, 1% for sorting substance-use dataset, and less than 1% was on the actual concatenation of the datasets.

Further analysis of the data integration of substance-use and Naloxone administration dataset in Figure 4.10 shown alcohol as the major substance but decreased from 93% in 2011 to 50 % in 2021 while other substance-use (e.g., Opioid) increased consistently from less than 5% in 2011-2019 to over 15% in 2020. Moreover, Figure 4.11 shows that over 50% of users are males. While crystal melt has been increased

since 2018, more than 50% of meth users are females. Naloxone administration more than doubled in 2016 with most cases requiring single dose. In extreme cases, a single person required over 10 dozes of the antagonist drug to reduce the effect of drug overuse.

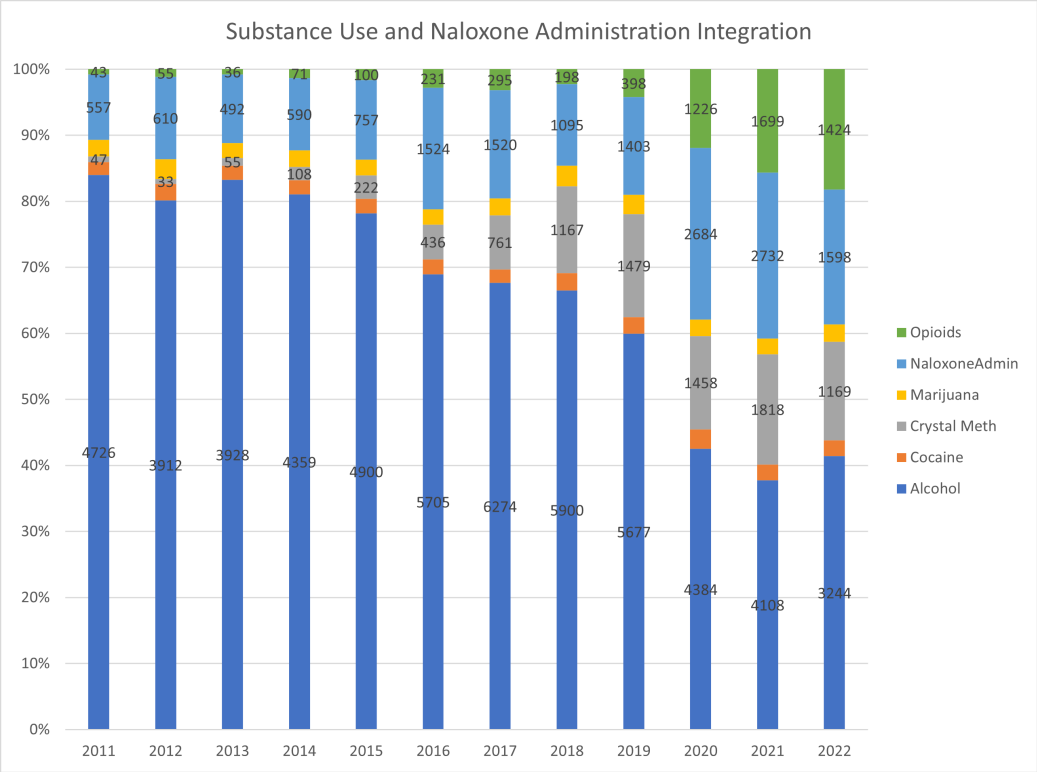


Figure 4.10: Substance-use and naloxone administration data integration

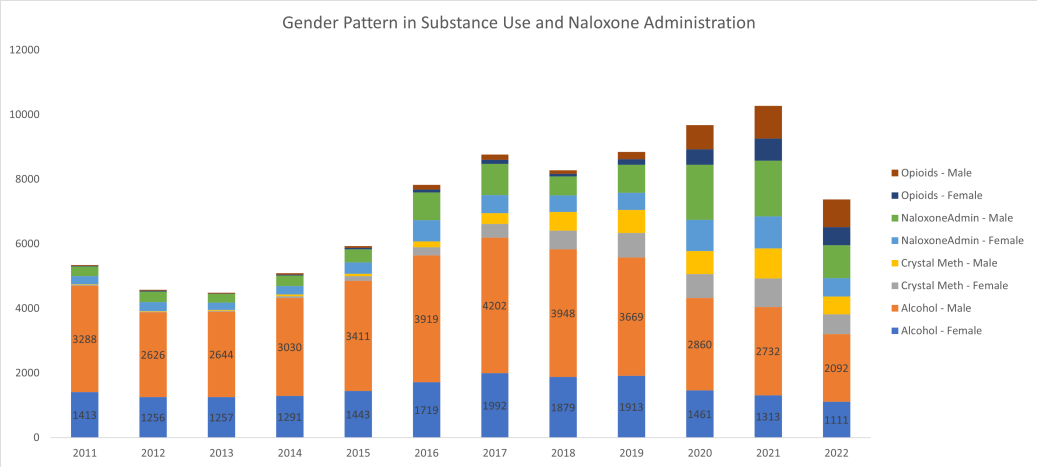


Figure 4.11: Gender pattern substance-use and naloxone administration

4.5.2 Analytical Evaluation on Spatial Representative Points

In this section, **analytical evaluation** on different spatial representative points for the HSTM is presented. Specifically, the following are the complexity analysis of representative points:

- Random representative point: This is a loosely computed point selected at random. Random operation can be seen as a GET operation from a list with linear computation overhead. The time complexity is estimated to be linear $O(1)$. if we consider loading the entire data set and traversing to pick a point, the space complexity should be the same as the number of points in the dataset given as $O(n)$, however, using dynamic approach to pick a point without traversing the entire dataset should give a linear space complexity of $O(1)$.

- Centroid: The simple calculation of a centroid is the average of all x coordinates and y coordinates. This is a simple arithmetic operation requiring $O(1)$ in the best-case scenario. The space requirement is dependent on the number of points in the dataset in the worst case given as $O(n)$.

- Medroid: The calculation for medroid involves distance calculation of point closest to the centroid, hence, it has two steps. First, the centroid which is a simple arithmetic average linear time complexity of $O(1)$, then the distance

between each of the point and the calculated centroid requires all x-coordinates and y-coordinates. Hence the time complexity is dependent on the number of points in the dataset given as $O(n)$. Thus, the medroid time complexity can be approximated as $O(n + 1)$. The space requirement includes all points in the dataset for the calculation of the minimum distance to the centroid and selecting one out of all point is a linear operation, thus the space complexity can be approximated as $O(n)$.

- Convex hull: The time complexity for calculating convex hull for a given set of coordinates largely depends on the size of xy points in the dataset. There are several implementations of convex hull [Ken23, NSR+19, BDH96, VGO+20]. For instance, the existing implementation in Python using the scientific/technical computing library SciPy [VGO+20]. The implementation uses quick hull algorithm designed by Barber et al. [BDH96] based on divide and conquer approach with time complexity reported as $O(n \log n)$ where n is the number of (x, y)-coordinates in the dataset. The space complexity also depends on the size of the coordinates of the dataset and the number vertices of the convex hull which is expected to be less than the size of the dataset given as $O(n+v)$ and selecting the centroid of the convex hull as the representative point is a linear operation $O(1)$ and insignificant.

- Bounding box: The time complexity for calculating bounding box for a given set of coordinates has two steps: (a) calculating the minimum and maximum for xy coordinates as given in Eq. (4.12). This arithmetic operation requires the comparing the entire point using divide and conquer approach and it is estimated to be $O(n \log n)$, then (b) calculating the centroid of the bounding box which is a linear operation $O(1)$ using simple average, thus the time complexity is $O(n \log n + 1)$. The space complexity also requires space for the entire x, y coordinates for the first operation, that is $O(n)$ and the second part requires just a linear space for selecting the centroid of the box. Thus, the space complexity is $O(n + 1)$.

Table 4.11: Spatial representative point complexity analysis

Method	Time complexity	Space complexity
Random point	$O(1)$	$O(n)$, $O(1)$
Centroid	$O(1)$	$O(n)$
Medroid	$O(n+1)$	$O(n)$
Convex hull	$O(n \log n + 1)$	$O(n+v+1)$
Bounding box	$O(n \log n + 1)$	$O(n+1)$

The summary for the complexity analysis is presented in Table 4.11. Note that the model presented here is a generic model and presents endless world of analytics variation of spatial and temporal concepts. For instance, there are other variations of the model that are possible and applicable in real-life such as:

- Rather than starting from temporal hierarchy, we may start with location hierarchy if we are interested in specific location of interest and then select the temporal hierarchy appropriate for the (privacy) level as shown in Figure 4.12. This is similar to a SQL query:

  > Select l_i, t_i, aggregate($a_1, a_2..a_n$) group by l_i, t_i where l_i is the specific location hierarchy t_i is the specific temporal hierarchy and $a_1, a_2..a_n$ are the other grouping reference attributes

- We may as well consider each component separately depending on requirements:
  - Apply only temporal hierarchy to any data where time is of utmost importance.
  - Apply location hierarchy when emphasis of data analytics is on location.
  - Use the representative point, either random, centroid or the hybrid for spatial representation.

4.6 Summary

After the design and implementation of an open data lake architecture with a metadata collection framework in the previous chapter, this chapter designed and

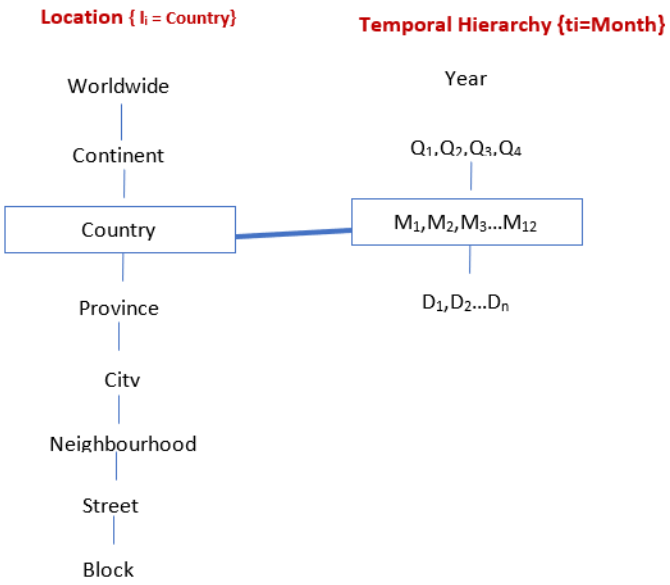


Figure 4.12: Location-temporal hierarchy

implemented (a) a hierarchical temporal model and (b) a hierarchical spatial model. The former captures temporal information of data points at different temporal granularities (e.g., century, decade, year, month, week, day, hour, minute, second), whereas the latter captures spatial information of data points at different spatial granularities (e.g., continent, country, province, city/town, street). Moreover, this chapter described the non-trivially integration and linkage of these two hierarchical models into a hierarchical spatio-temporal model (HSTM) to capture spatio-temporal information of data points at different spatial and/or temporal granularities. Experimental evaluation on four real-world open data shows the practicality of the HSTM. Moreover, the chapter explored different spatial representative points for representing spatial locations of data points in temporal sequences in the HSTM. Finally, the chapter presented the analytical evaluation discussing the time and space complexity of these representative points.

Chapter 5

Hierarchical Spatio-temporal Privacy Preserving Model (HSTPPM)

After designing and implementing a hierarchical spatio-temporal model (HSTM) in an open data lake architecture with a metadata collection framework, this chapter explores how to preserve privacy in the HSTM. Specifically, let us non-trivially incorporate privacy preservation into the HSTM to form hierarchical spatio-temporal privacy preserving model (HSTPPM). The key contributions of this chapter include the design and implementation of the HSTPPM, as well as its adaption and practical applications for privacy-preserving publishing and COVID-19 contact tracing.

5.1 Privacy Preservation by the HSTPPM

Given a dataset D (e.g., parking, health or airline check-in dataset), the HSTPPM creates (a) temporal hierarchy for date generalization, (b) extended temporal hierarchy for time generalization, and (c) extended hierarchy for spatial data for location generalization. In the temporal hierarchy, for each item (e.g., each parking violation, each contact tracing data, airline check-in transaction, etc.), the HSTPPM select a temporal hierarchy, for preserving privacy of individuals (e.g., parking violators,

COVID-19 patient). Then, the model counts the frequency of the aggregated data, time and location to report the aggregated information for big data analytics. While preserving privacy of temporal big data, the model also maintains the utility of the data by providing sufficient information for the analytics. A skeleton of the model for temporal and location hierarchy is presented in Algorithms 5.1 and 5.2 is applicable to the three models(when spatial coordinates are available). Differential privacy may be added to the temporal representative point when necessary (because of its privacy assurance) using the assumption that a randomized mechanism A satisfies ϵ-differential privacy on spatio-temporal representative point provided that any location output z_t from a spatio-temporal dataset x_t and a neighboring spatio-temporal dataset x_t^* obtained by the addition or removal of a record is not significantly different.

Algorithm 5.1 Hierarchical temporal location model

```
1: Input: Dataset D
2: Output: Privacy-preserving dataset D'
3: Date ← create temporal hierarchy
4: Location ← create location hierarchy
5: for each temporal hierarchy do
6:     for each location hierarchy in D do
7:         group by Time, Location
8:         compute frequency count
```

Algorithm 5.2 Spatio-temporal hierarchy with representative point

```
1: Input: Spatio-temporal Dataset D
2: Output: Privacy-preserving Spatio-temporal dataset D'
3: Date ← create temporal hierarchy
4: for selected temporal hierarchy do
5:     group by Location, attribute 1, attribute2, ..., attribute n
6:     compute frequency count
7:     compute representative point for the (x, y) group
```

The solution is guaranteed to preserve the privacy of both the actual timestamp

and location by using:

- The temporal hierarchy to preserve the privacy of the actual timestamp, and
- The spatio-temporal representative point to preserve the privacy of the actual location.

Example 5.1. *Let us illustrate Algorithm 5.1 by considering COVID-19 contact tracing route data between 25-28 February 2020, with 12 sample patient route information presented in Table 5.1. All patient route records were from the South Korean capital city of Seoul, in the district of Gangnam-gu. The first three records occurred in 25 February 2020 when patients used public transportation. Among the next six records for 26 February 2020, three of them dined at restaurants, two visited a cafe, and the last record represents a visit to a pharmacy. The final three records occurred in February 28 when patients also dined at restaurants.*

The algorithm builds a temporal hierarchy, groups these 12 records by day, district, city, type of establishment (e.g., public transportation, restaurant), and generates a coordinate list for the resulting group as shown in Table 5.2.

Afterwards, it computes the spatio-temporal representative points for these four groups using the medroid or centroid approach. For any group not satisfying the minsup threshold (e.g., groups 3 and 4 do not satisfy minsup of 3 records), the type is changed to "others", re-aggregate and test again. Specifically, for groups 3 and 4, the cafe and the pharmacy are grouped as "others" and recalculates their spatio-temporal representative point for the resulting group. The final output of the sample dataset is presented in Table 5.3 where one point is representing each group.

Thus, the model provides protection for the patients such that their information is hidden in space and time. Individual patient could be traced in the original unprotected dataset by an adversary with background knowledge given the time and location visited using the spatial and temporal attributes. For instance, a patient trajectory visiting different locations such as store and restaurants by public transportation before finally tested in the hospital. However, with the proposed solution, the actual patient identity is unknown.

Table 5.1: South Korean patient route data

ID	**Date**	**District ("gu")**	**City**	**Type**	**Latitude**	**Longitude**
68	2020-Feb-25	Gangnam-gu	Seoul	public transportation	37.493601	127.079526
86	2020-Feb-25	Gangnam-gu	Seoul	public transportation	37.487468	127.101324
93	2020-Feb-25	Gangnam-gu	Seoul	public transportation	37.514236	127.031593
63	2020-Feb-26	Gangnam-gu	Seoul	restaurant	37.499907	127.037393
66	2020-Feb-26	Gangnam-gu	Seoul	restaurant	37.504689	127.043847
86	2020-Feb-26	Gangnam-gu	Seoul	restaurant	37.516567	127.021503
66	2020-Feb-26	Gangnam-gu	Seoul	cafe	37.505153	127.044054
86	2020-Feb-26	Gangnam-gu	Seoul	cafe	37.522466	127.037943
93	2020-Feb-26	Gangnam-gu	Seoul	pharmacy	37.509681	127.032382
86	2020-Feb-28	Gangnam-gu	Seoul	restaurant	37.516567	127.021503
93	2020-Feb-28	Gangnam-gu	Seoul	restaurant	37.514236	127.031593
105	2020-Feb-28	Gangnam-gu	Seoul	restaurant	37.504356	127.043652

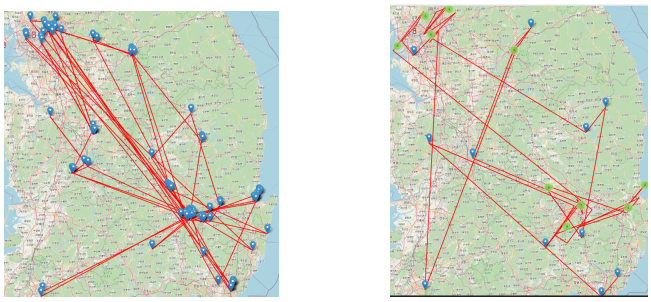
Table 5.2: Location set aggregated from the temporal hierarchy

Group#	**#rec in group**	**Date**	**District**	**City**	**Type**	**Location set**
1	3	2020-Feb-25	Gangnam-gu	Seoul	public transportation	{(37.493601, 27.079526), (37.487468, 127.101324), (37.514236, 127.031593)}
2	3	2020-Feb-26	Gangnam-gu	Seoul	restaurant	{(37.499907, 127.037393), (37.504689, 127.043847), (37.516567, 127.021503)}
3	2	2020-Feb-26	Gangnam-gu	Seoul	cafe	{(37.505153, 127.044054), (37.522466, 127.037943)}
4	1	2020-Feb-26	Gangnam-gu	Seoul	pharmacy	{(37.509681, 127.032382)}
5	3	2020-Feb-28	Gangnam-gu	Seoul	restaurant	{(37.516567, 127.021503), (37.514236, 127.031593), (37.504356, 127.043652)}

Table 5.3: Spatio-temporal representative point

Group#	**#rec in group**	**Date**	District	City	**Type**	**Spatio-temporal rep. pt.**
1	3	2020-Feb-25	Gangnam-gu	Seoul	public transportation	(37.498435, 127.070814)
2	3	2020-Feb-26	Gangnam-gu	Seoul	restaurant	(37.507054, 127.034247)
3'	3	2020-Feb-26	Gangnam-gu	Seoul	**others**	(37.507054, 127.038126)
5	3	2020-Feb-28	Gangnam-gu	Seoul	restaurant	(37.511719, 127.032249)

Moreover, the notion of near object relationship is extended to privacy by grouping all patients and their activities under temporal and spatial correlation. For instance, patients route trace Figure 5.1(a) shows all locations visited by 69 patients on 17 February 2020, with 128 points. The version of the same patient route data after



(a) Patient trace Feb 17 (b) Protected trace Feb 17

Figure 5.1: SKCOVID-19 data before and after protection

applying the proposed solution is shown in Figure 5.1(b), where no patient information is revealed. Privacy preserved data are represented by 53 temporal representative points such that the knowledge of any adversary is limited to the trends within the aggregated time series data. The grouping of patients using temporal hierarchy still provides essential route information necessary for contact tracing without relating the information to any specific patient.

5.2 Privacy-Preserving Publishing by the HSTPPM

To support privacy-preserving publishing of by-law enforcement data, one can apply temporal and location hierarchy in the HSTPPM to preserve the privacy of the actual timestamp and location of the trajectory data (i.e., temporal sequences of spatial data points). As an example of by-law enforcement data, parking violation tickets may capture the actual timestamp and location of the violations. These data can be preprocessed. For instance, the actual timestamp can be aggregated into two time partitions—namely, day and night times. These can then be aggregated upwards in the temporal hierarchy into day, week, month, quarter, year, etc. Similarly, the location can be aggregated into a small geographic unit (e.g., street, census block), which can then be aggregated upwards in the spatial hierarchy into neighborhood, district, city/town, province/state, country, etc. As a preview, case studies on trajectory data about parking violation tickets in (a) Toronto, ON, Canada and (b) Buffalo, NY, USA will be studied in Sections 5.4.2 and 5.4.3, respectively.

5.3 Contact Tracing by the HSTPPM

To support COVID-19 contact tracing, one can apply temporal and location hierarchy in the HSTPPM to preserve the privacy of the COVID-19 contact tracing data (i.e., temporal sequences of spatial data points). These data can be preprocessed. For instance, temporal information can be aggregated into day or week, which can then be aggregated upwards in the temporal hierarchy into month, quarter, year, etc. Similarly, spatial information can be aggregated into a neighborhood, district, or city/town, which can then be aggregated upwards in the spatial hierarchy into *health authority* or *health region* (e.g., Winnipeg Regional Health Authority, Vancouver Coastal Health region, Toronto Central Local Health Integration Network (LHIN)). Along this direction, health authority/region can be further aggregated upwards in the spatial hierarchy into province/state, country, etc. As a preview, case studies on COVID-19 contact tracing data will be studied in Section 5.4.4.

To elaborate, ***week***was added to the temporal hierarchy. This is necessitated by the nature of the COVID-19 infectious diseases spreading rapidly during the study period of January to March 2020. It is obvious, that contact tracing data publishing can be for social good helping people around the cluster of exposure to go for testing and self isolate safely and thereby slow down the spread of the COVID-19.

In addition, the concept of minimum support (which is defined as the minimum count of occurrence at a particular location on specific temporal hierarchy) was added to the algorithm. The users can change the minimum support threshold depending on the requirement of the application and use:

$$minsup = \min(l, t, s) \tag{5.1}$$

where l is the location hierarchy, t is the temporal hierarchy and s is the count of patient in l at time t. Any place not meeting the minimum support threshold is reclassified as "*others*".

5.3.1 DP-HSTPPM: Differential Privacy Based HSTPPM

A way to preserve privacy (DP) in the HSTPPM is to add Laplace differential privacy noise to the HSTM and result in DP-HSTPPM. More specifically, the task for DP-HSTPPM has the following steps:

- Add noise to individual point varying the noise parameter,
- Aggregate the data by province, city and infection route type,
- Varying the temporal hierarchy by ***day, week*** and ***month***;
- Calculate medroid for each temporal hierarchy; and
- Calculate the root mean square error (RMSE).

The probability density function for this Laplace mechanism having mean μ and epsilon ϵ level of noise is given as:

$$f(x \mid \mu, \epsilon) = \frac{1}{2\epsilon} e^{-\left|\frac{x-\mu}{\epsilon}\right|} \tag{5.2}$$

By adding Laplace differential privacy noise, at any level t of the temporal hierarchy, a randomized mechanism A satisfies ϵ-differential privacy on spatio-temporal representative point provided that any location output z_t from a spatio-temporal dataset x_t and a neighboring spatio-temporal dataset x_t^* obtained by the addition or removal a record is not significantly different:

$$\frac{Pr\left(A(x_t = z_t)\right)}{Pr\left(A(x_t* = z_t)\right)} \leq e^{\epsilon} \tag{5.3}$$

5.3.2 GAN-HSTPPM: Generative Adversarial Network (GAN) Based HSTPPM

Another way to preserve privacy in the HSTPPM is to use the GAN machine learning privacy preserving method on the contract tracing data. The resulting GAN-HSTPPM architecture contains the following components:

- *Patient trajectory discriminator*, which is a multi-layer perceptron network (MLP) comprising of input layer, three hidden layers and an output layer as shown in Figure 5.2.

- *Patient trajectory generator*, which is similarly setup as a MLP comprising of input layer, two hidden layers and an output layer as shown in Figure 5.3 varying the activation function and using linear activation function in the output layer.

Within it, different activation functions—such as rectified linear unit function (ReLU), sigmoid, and/or tanh—can be applied:

- rectified linear unit function (ReLU) is defined as:

$$ReLU(x) = \max(0, x) \tag{5.4}$$

- sigmoid activation function is defined as:

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{5.5}$$

- tanh activation function is defined as:

$$tanh(x) = \frac{\epsilon^{x} - \epsilon^{-x}}{\epsilon^{x} + \epsilon^{-x}} \tag{5.6}$$

- linear activation function is defined as:

$$linear(x) = ax + b \tag{5.7}$$

  where a and b are some constants.

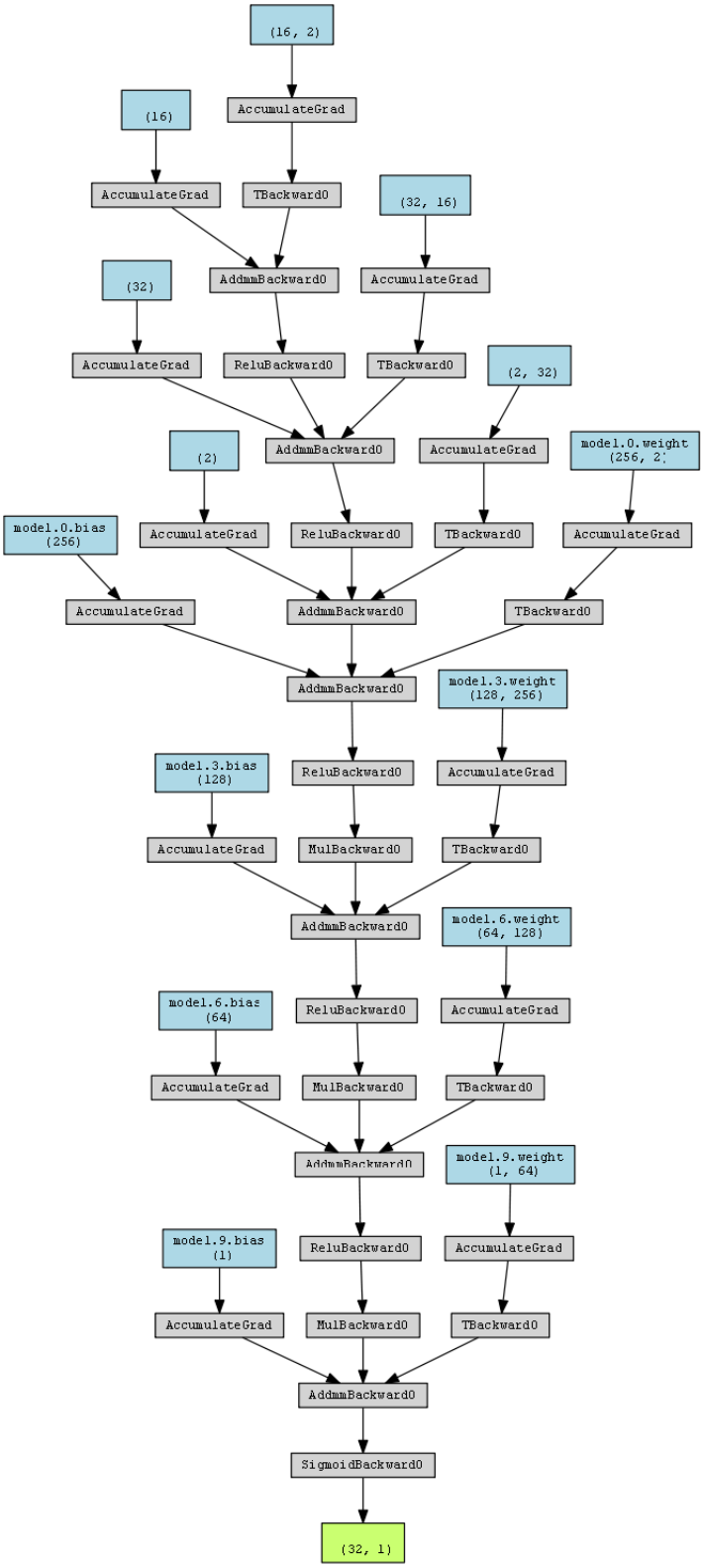


Figure 5.2: GAN patient discriminator architecture

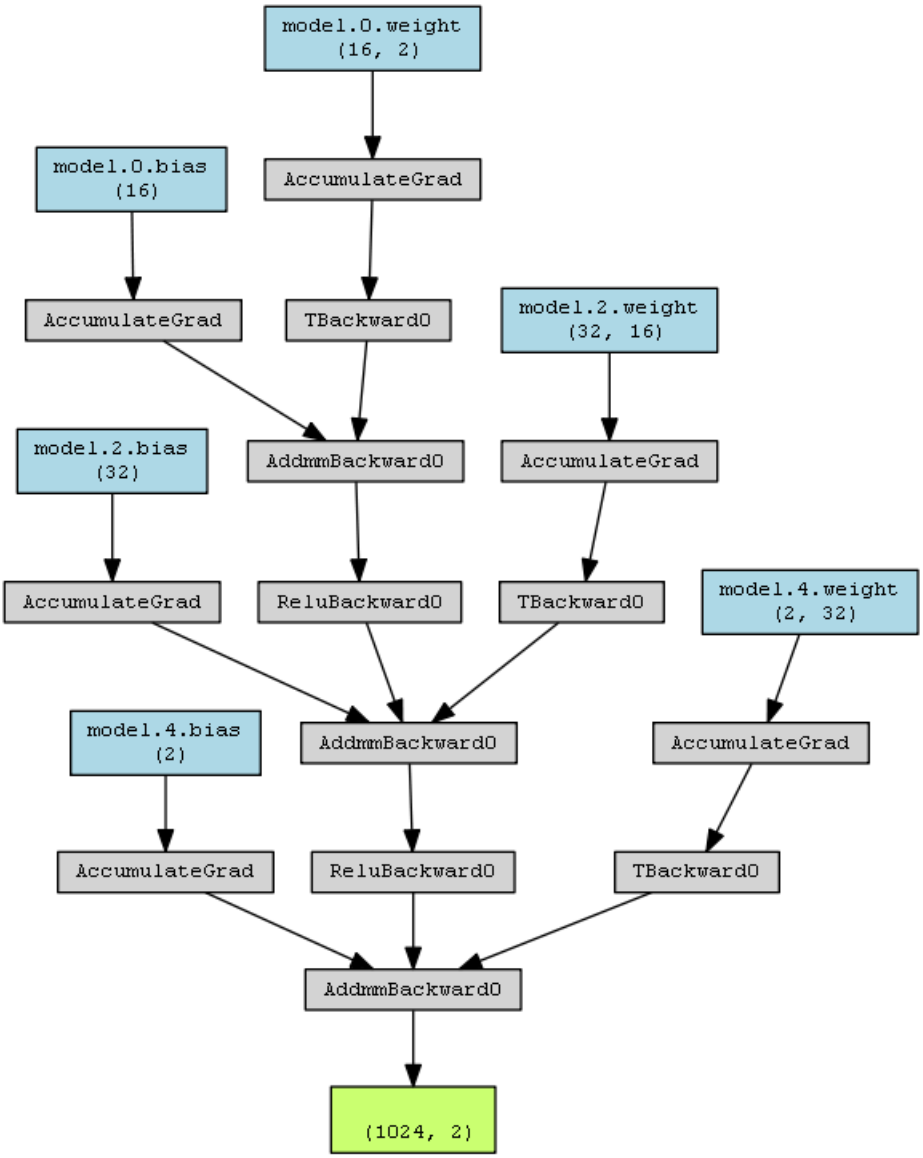


Figure 5.3: GAN patient private trajectory generator architecture

When training the generator, for each epoch, a batch of dataset is taken from the real dataset. The same size is generated through the generator. The input to the generator is random numbers from standard normal distribution with a mean $\mu = 0$ and variance = 1. When training the discriminator, the input to the discriminator is the combined *real data and generated data* for each batch size.

5.4 Evaluation

5.4.1 Evaluation on the HSTPPM

In terms of **setup**, implementation of the HSTPPM was done in Python on Spyder scientific development environment. Experimental evaluations were conducted on a laptop computer with 2.6 GHz Intel(R) core^{TM} i7 64-bit operating system and 8 GB installed memory.

In terms of **evaluation metrics**, measurements recorded include spatial utility, root mean square error, spatial compression ratio, change in file size, and aggregate queries . Details of these evaluation metrics are explained below:

- **Spatial utility**, which measures the utility of the temporal representative point by computing the distance between the released location and true location using great circle distance. The great circle distance is defined as:

$$D\left(x,y\right)=2\ \arcsin\sqrt{\sin^2\left(\frac{x_1-y_1}{2}\right)+\cos(x_1)\cos(y_1)\sin^2\left(\frac{x_2-y_2}{2}\right)} \tag{5.8}$$

- **Root mean square error (RMSE)**, which is a generic risk metric that measure the magnitude of the error loss function:

$$RMSE=\sqrt{\sum\left((x_1,y_1)-(x_2,y_2)\right)^2} \tag{5.9}$$

Generally, aggregation reduces the number of records in a dataset and as a result the file size also reduces. In addition, the temporal representative point also reduces the number of spatial information released for the dataset. Hence, spatial compression ratio and file size compression ratio are measured and recorded:

- **Spatial compression ratio (SCR)**, which is defined as a ratio of distinct spatial points in anonymized dataset to the spatial points in original dataset:

$$SCR\left(D,D\text{'}\right)=\frac{count\left(P\right)}{count\left(P'\right)} \tag{5.10}$$

where P and P' are the numbers of points in database D and D'.

- **Data compression ratio**, which is defined as the ratio of the anonymized dataset D' to the original dataset D at a specific temporal hierarchy:

$$DCR\,(D, D') = \frac{D'}{D} \tag{5.11}$$

Note that the size and number of attributes from an anonymized framework may be different from the original dataset due to aggregation. In the proposed model, the size of the anonymized dataset is quite smaller than the original dataset due to temporal hierarchy that support aggregation of records. The file compression ratio FCR is defined as:

- **File compression ratio**, which is defined as the ratio of the anonymized dataset D' to the original dataset D as follows:

$$FCR\,(D, D') = \frac{size\,(D)}{size\,(D')} \tag{5.12}$$

- **Aggregate queries** by varying the temporal hierarchy level and aggregation parameter. When aggregated by different location hierarchy such as country, city, district and street levels.

5.4.2 Evaluation Case Study 1 on the HSTPPM: Privacy-Preserving Publishing of By-Law Enforcement Data in Toronto

Toronto parking ticket[1], which captures non-identifiable information relating to parking tickets for the city of Toronto in the Canadian province of Ontario. This 1.47 GB dataset contains 11,096,751 complete violation records for a 5-year period from January 2014 to December 2018, issued by Toronto Police Services (TPS) personnel as well as persons certified and authorized to issue tickets by TPS. Table 5.4 provides some detailed descriptions of the dataset. With this description, readers can

[1]https://open.toronto.ca/dataset/parking-tickets/

interpret data about infraction. For example, they can observe that Ticket ***91886 was issued on 29 September 2018, for an infraction with infraction code 2 for parking longer than 3 hours—with a set fine amount of $15. The infraction occurred at 22:33 on the west side of Rumney Road and north of Trenton Avenue. The vehicle was registered with an Ontarian plate.

Table 5.4: Toronto parking ticket data description

Attribute	Description	Data Type
tag number masked	Masked ticket ID	text
date of infraction	The date when the infraction (e.g., parking violation) occurred	date
infraction code	Numeric code for the infraction	text
infraction description	A short description of the infraction	text
set fine amount	Amount of set fine associated with the infraction	float
time of infraction	The time when the infraction occurred	time
location1	Location proximity code (e.g., at, near, opposite, rear of, north side, south of, etc.)	text
location2	Street address	text
location3	Optional location proximity code	text
location4	Optional street address	text
province	Province code of vehicle license plate (e.g., ON for Ontario)	text

The goals for this evaluation are to (a) applies temporal and location hierarchy to preserve the privacy of the actual timestamp and location of the trajectory data, and to (b) demonstrate how the HSTPPM helps in data compression. To do so, first, the data were preprocessed by partition the time hierarchy into two levels:

1. Daytime capturing tickets issued from the 12-hour periods of 06:00-18:00;

2. Nighttime capturing tickets issued from the 12-hour periods of 18:00-06:00.

Here, street is a geographic unit used for the Toronto parking tickets. The block information was created by partitioning each street into group of hundred addresses. Three experiments were conducted to measure (a) the impact of variation in hierarchy level, (b) the reduction in number of records in the dataset (i.e., data compression ratio) at each temporal hierarchy level, and (c) the unique record percentage for each level.

Experiment 5.1. *Held the location details constant, generalized the time component, and varied the temporal hierarchy:*

SELECT province, street, block, time, (T1|T2|T3|T4), COUNT(*)
FROM ...
WHERE ...
GROUP BY province, street, block, time (T1|T2|T3|T4)

where T1, T2, T3 and T4 are the temporal hierarchy level representing day, month, quarter and year, respectively.

Observation 5.1. *For Toronto parking ticket, the number of records at each temporal hierarchy for Experiment 5.1 is presented in Figure 5.4. The number of records in the aggregated dataset reduces as we move up the hierarchy level.*

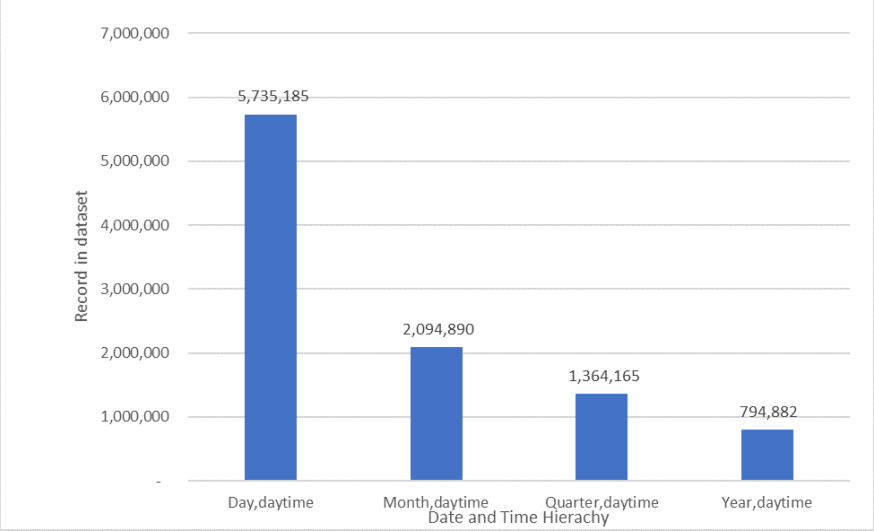


Figure 5.4: Impact of variation in hierarchy level (Experiment 5.1: date+time generalization): Toronto data

With time generalization (to the daytime and nighttime level), the number of aggregated data drops from 5,735,185 aggregated daily counts to 2,094,890 aggregated monthly counts, then to 1,364,165 aggregated quarterly counts, and finally to 794,882 aggregated yearly counts of infractions in Toronto.

Experiment 5.2. *Location generalization and temporal hierarchy* without *the time generalization:*

SELECT province, street, block, (T1|T2|T3|T4), COUNT(*)
FROM ...
WHERE ...
GROUP BY province, street, block, (T1|T2|T3|T4)

Observation 5.2. *For Toronto parking ticket, as shown in Figure 5.5, the number further reduces in Experiment 5.2 because of the suppression of the time generalization. By suppressing the time generalization (i.e., not distinguishing the daytime and nighttime data), the number of aggregated data reduces further. Specifically, it starts with 5,364,835 aggregated daily counts, which drops to 1,748,291 aggregated monthly counts, then to 1,111,379 aggregated quarterly counts, and finally to 632,295 aggregated yearly counts of infractions in Toronto. This number of aggregated yearly counts is less than 80% of that with time generalization.*

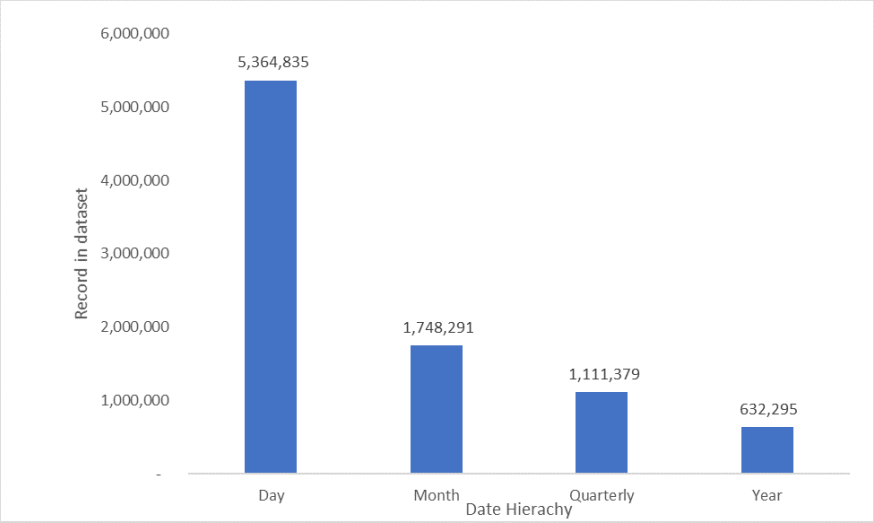


Figure 5.5: Impact of variation in hierarchy level (Experiment 5.2: date generalization): Toronto data

Experiment 5.3. *Street location generalization and varied temporal hierarchy.*

Observation 5.3. *For Toronto parking ticket, the location generalization to street level further reduces the number in Experiment 5.3 as shown in Figure 5.6. When incorporating location generalization (to the street level), the number of aggregated data reduces further. Specifically, it drops to 4,232,982 aggregated daily counts, then to 1,214,532 aggregated monthly counts, further 760,382 aggregated quarterly counts, and finally to 426,096 aggregated yearly counts of infractions in Toronto. This number of aggregated yearly counts is less than 68% and less than 54% of those in Experiment 5.2 and Experiment 5.1, respectively, which both without location generalization. Experiment 5.3 shows the benefits of location generalization. These aggregated counts for the three experiments are summarized in Table 5.5.*

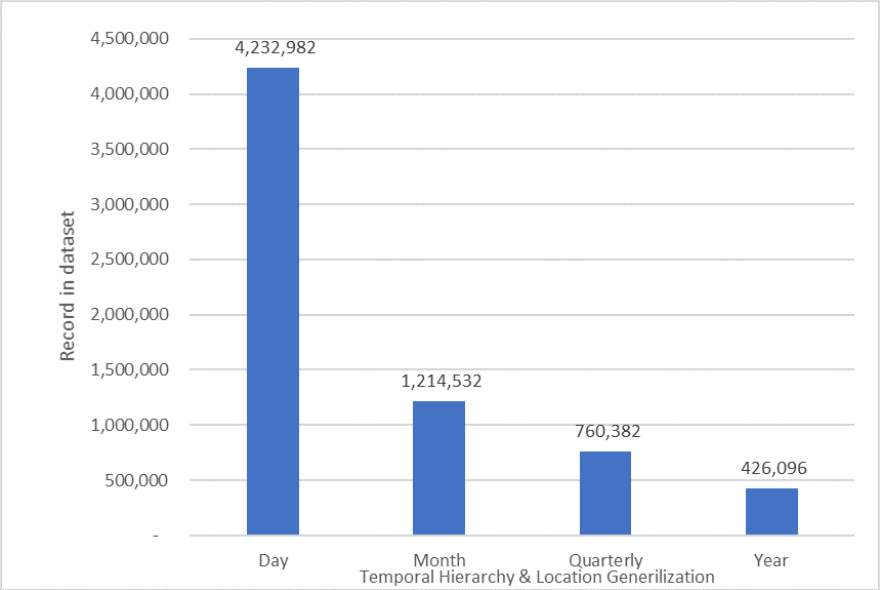


Figure 5.6: Impact of variation in hierarchy level (Experiment 5.3: date+location generalization): Toronto data

Table 5.5: Impact of variation in hierarchy level: numbers of aggregated counts in Toronto data

Hierarchy Level	**Experiment**		
	5.1	**5.2**	**5.3**
Year	794,882	632,295	426,096
Quarter	1,364,165	1,111,379	760,382
Month	2,094,890	1,748,291	1,214,532
Day	5,735,185	5,364,835	4,232,982

Experiment 5.4. *Note that the unique record is the frequency count of record where the count of ticket remains one despite aggregation, which can be obtained by running a SQL query:*

COUNT(*) ... HAVING COUNT(ticket) < 2

Observation 5.4. *For Toronto parking ticket, Table 5.6 and Figure 5.7 show that the* **unique record percentage** *drops from 30 percent to 1.8 percent when varying the hierarchy level.*

Table 5.6: Unique record percentage: Toronto data

Hierarchy Level	**Experiment**		
	5.1	**5.2**	**5.3**
Year	3.53%	2.74%	1.80%
Quarter	6.11%	4.82%	3.21%
Month	9.52%	7.65%	5.13%
Day	33.30%	30.08%	21.70%

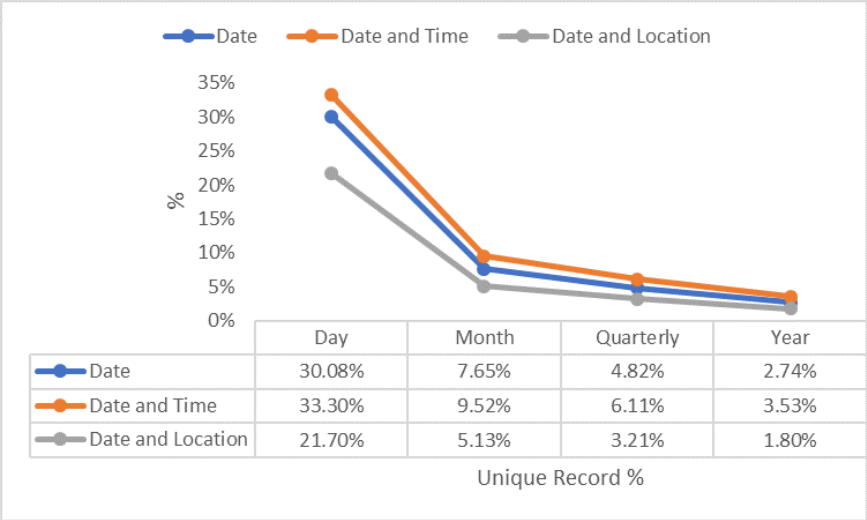


Figure 5.7: Unique record percentage: Toronto data

Observation 5.5. *For Toronto parking ticket, as shown in Figure 5.8, aggregating to the lowest temporal hierarchy resulted in at least 50 percent* **data compression** *aside from preserving privacy of individual record.*

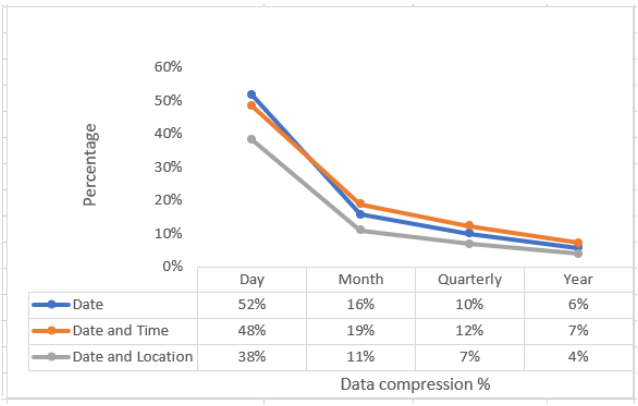


Figure 5.8: Data compression ratio: Toronto data

Experiment 5.5. *Furthermore,* ***count queries*** *was measured by performing series of count queries (histogram) over the aggregated dataset for each temporal hierarchy level to measure the utility of the anonymized published dataset.*

Observation 5.6. *For Toronto parking ticket, the result of the aggregate query over the protected and original dataset are the same because temporal hierarchy is essentially a bottom-up aggregation of record.*

Observation 5.7. *For Toronto parking ticket,* **query execution time** *for each experiment at various levels of the hierarchy is presented in Figure 5.9. The computed query result for the lowest level of the temporal hierarchy was used such that the original dataset is read only once. All other queries are computed from the lower level of the temporal hierarchy resulting in reduction of the query execution time.*

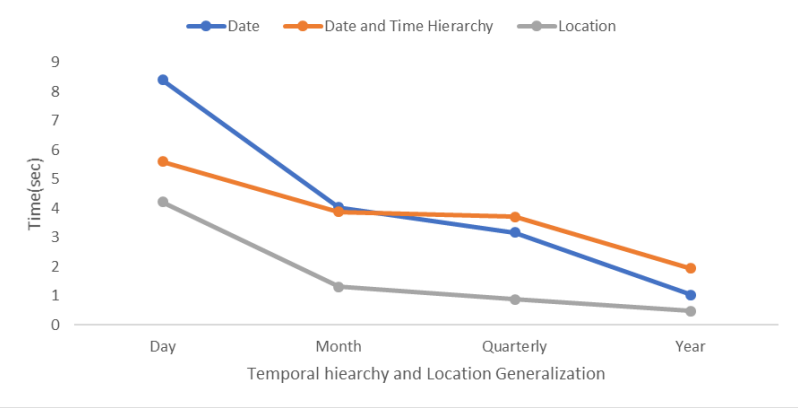


Figure 5.9: Query execution time: Toronto data

5.4.3 Evaluation Case Study 2 on the HSTPPM: Privacy-Preserving Publishing of By-Law Enforcement Data in Buffalo

.

To further evaluate the HSTPPM, the evaluation was repeated by applying it to the second dataset—namely, the Buffalo parking ticket dataset:

- **Buffalo parking ticket**[2], which captures parking tickets for the city of Buffalo in the New York state, USA. This 413 MB dataset contains 2,283,838 parking summonses for a 13-year period from January 2007 to December 2019, issued by the Division of Parking Enforcement, the Buffalo Police Department, and other agencies permitted to write summons on behalf of the City of Buffalo. Table 5.7 provides some detailed descriptions of the dataset. Readers can interpret that data about summon. For example, they can observe that summon A5508230 was issued on 01 January 2007 at 01:04 for a parking violation with its original fine amount. The parking violation occurred on Washington Street in Buffalo, NY, and located at (42.891484°N, 78.871482°W) within the US census block 1006.

Recall that census block is a geographic unit used by the US Census Bureau. Hence, census blocks are used for the Buffalo parking tickets.

Table 5.7: Buffalo parking ticket data description

Attribute	Description	Data Type
summon number	Unique ID for each parking summon/ticket	text
summon date	The issue date of the summon	date
violation time	The issue time of the summon	time
violation description	A description of the violation	text
original fine	The original fine associated with the parking summon	float
violation address number	The street number where the parking violation occurred	text
violation street	The street where the parking violation occurred	text
violation full address	The full street address where the parking violation occurred	text
city	The city where the parking violation occurred	text
state	The state where the parking violation occurred	text
latitude	The latitude where the parking violation occurred	float
longitude	The longitude where the parking violation occurred	float
location	The geographic location (latitude, longitude) where the parking violation occurred	point
council district	The council district where the parking violation occurred	text
police district	The police district where the parking violation occurred	text
census tract	The *census tract* (which is a generalization of census block groups) where the parking violation occurred	float
census block group	The *census block group* (which is a generalization of census blocks) where the parking violation	integer
census block	The census block where the parking violation occurred	integer

[2]https://data.buffalony.gov/Transportation/Parking-summones/yvvn-sykd

Observation 5.8. *For Buffalo parking data, the reduction in the number of records for each of the three experiments (Experiments 5.1–5.3) is shown in Figures 5.10, 5.11 and 5.12. The number of records in the aggregated dataset reduces as we move up the hierarchy level. Specifically, Figure 5.10 shows that, with time generalization, the number of aggregated data drops from 1,519,895 aggregated daily counts to 899,988 aggregated monthly counts, then to 646,959 aggregated quarterly counts, and finally to 411,032 aggregated yearly counts when we move up the temporal hierarchy level for day to month to quarter and finally to year.*

Similarly, Figure 5.11 shows that, without time generalization, the number of aggregated data drops further to 1,501,133 aggregated daily counts, and to 839,468 aggregated monthly counts, then to 579,298 aggregated quarterly counts, and finally to 347,334 aggregated yearly counts when we move up the temporal hierarchy level for day to month to quarter and finally to year.

Moreover, Figure 5.12 shows that, with location generalization, the number of aggregated data drops further to 1,117,809 aggregated daily counts, and to 442,392 aggregated monthly counts, then to 279,799 aggregated quarterly counts, and finally to 152,305 aggregated yearly counts when we move up the temporal hierarchy level for day to month to quarter and finally to year. This number of aggregated yearly counts is less than 44% and less than 37% of those in Experiments 5.2 and 5.1, respectively, which both without location generalization. Experiment 5.3 again shows the benefits of location generalization. These numbers are summarized in Table 5.8.

Table 5.8: Impact of variation in hierarchy level: numbers of aggregated counts in Buffalo data

Hierarchy Level	**Experiment**		
	5.1	**5.2**	**5.3**
Year	411,032	347,334	152,305
Quarter	646,959	579,298	279,799
Month	899,988	839,468	442,392
Day	1,519,895	1,501,133	1,117,809

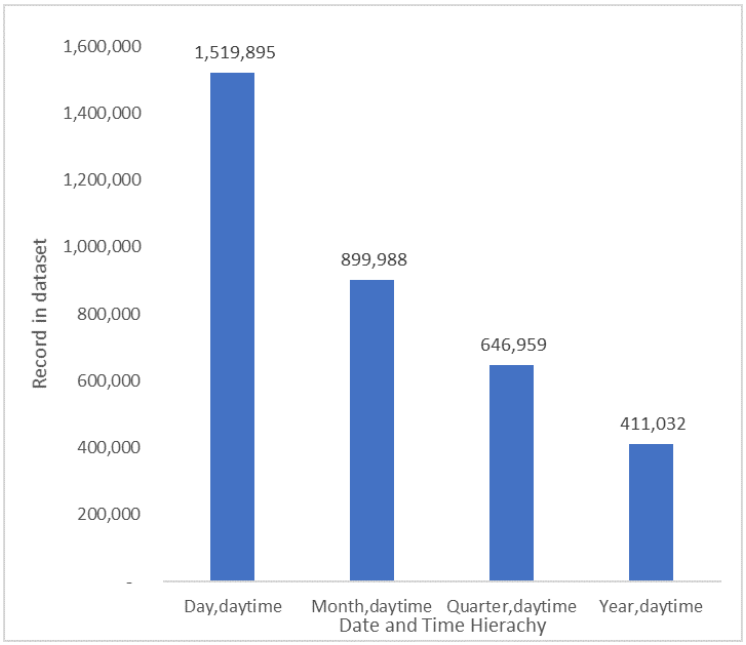


Figure 5.10: Impact of variation in hierarchy level (Experiment 5.1: date+time generalization): Buffalo data

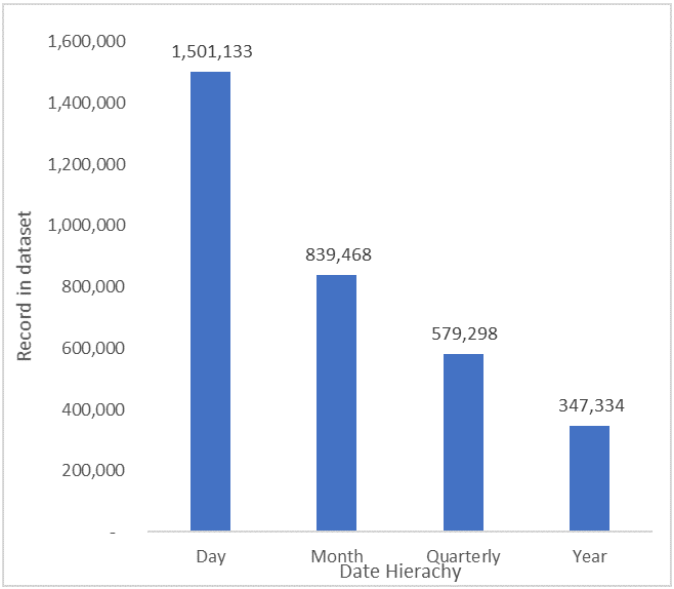


Figure 5.11: Impact of variation in hierarchy level (Experiment 5.2: date generalization): Buffalo data

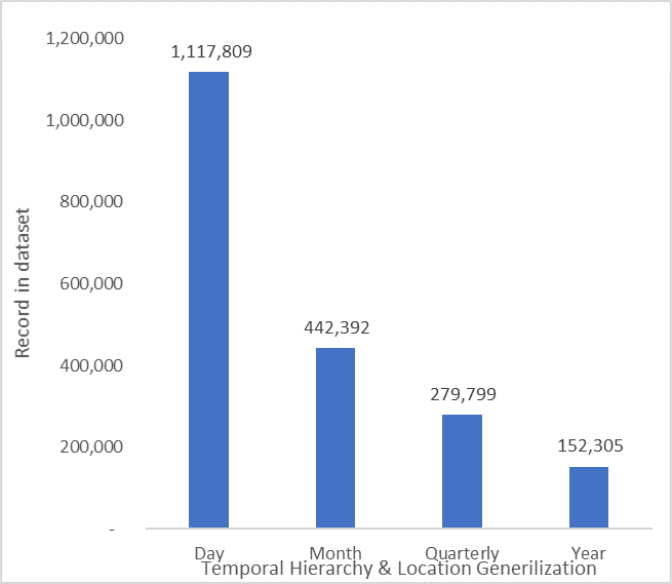


Figure 5.12: Impact of variation in hierarchy level (Experiment 5.3: date+location generalization): Buffalo data

Observation 5.9. *Experiment 5.4 was repeated for Buffalo parking data. Table 5.9 and Figure 5.13 show that the unique percentage drops from 48 percent (for the lowest level of temporal hierarchy) to 3 percent (for the year level) for the Buffalo dataset. This further shows that adding differential privacy Laplace noise may be necessary for some big data privacy preserving tasks.*

Table 5.9: Unique record percentage: Buffalo data

Hierarchy Level	**Experiment**		
	5.1	**5.2**	**5.3**
Year	10%	8%	3%
Quarter	16%	14%	6%
Month	24%	22%	10%
Day	49%	48%	31%

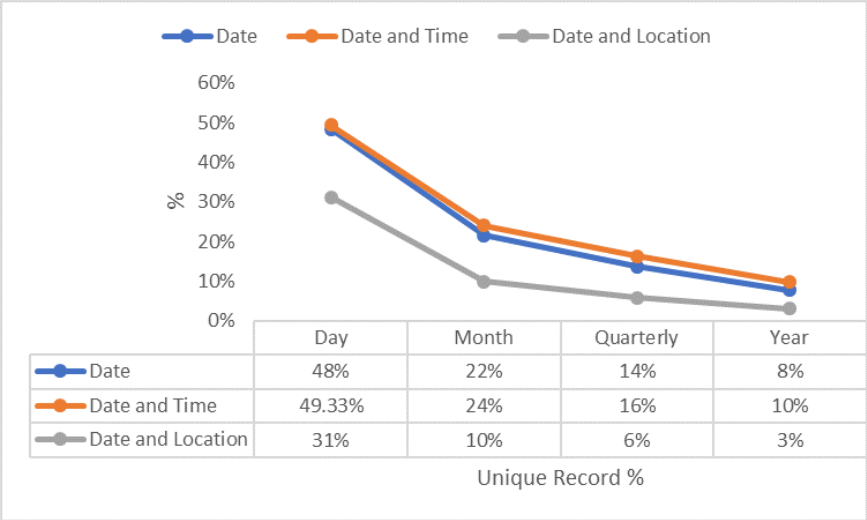


Figure 5.13: Unique record percentage: Buffalo data

Observation 5.10. *For Buffalo parking data, as shown in Figure 5.14, aggregating to the lowest temporal hierarchy resulted in at least 50 percent data compression aside from preserving privacy of individual record. See Table 5.10.*

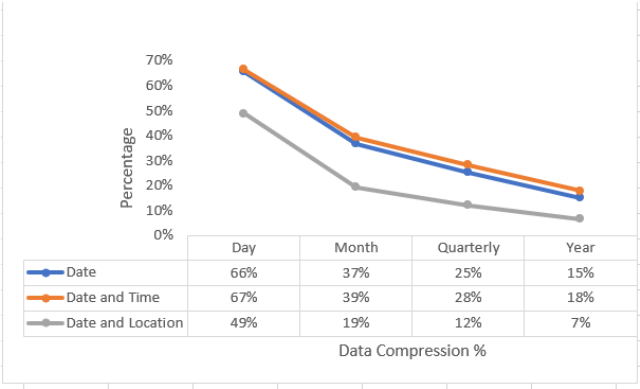


Figure 5.14: Data compression ratio: Buffalo data

Table 5.10: Data compression ratio: Buffalo data

Hierarchy Level	Experiment		
	5.1	5.2	5.3
Year	18%	15%	7%
Quarter	28%	25%	12%
Month	39%	37%	19%
Day	67%	66%	49%

Observation 5.11. *Experiment 5.5 was repeated as well for Buffalo parking data. The result of the aggregate query over the protected and original dataset are the same because temporal hierarchy is essentially a bottom-up aggregation of record.*

Observation 5.12. *For Buffalo parking ticket,* **query execution time** *for each experiment at various levels of the hierarchy is presented in Figure 5.15. The computed query result for the lowest level of the temporal hierarchy was utilized such that the original dataset is read only once. All other queries are computed from the lower level of the temporal hierarchy resulting in reduction of the query execution time.*

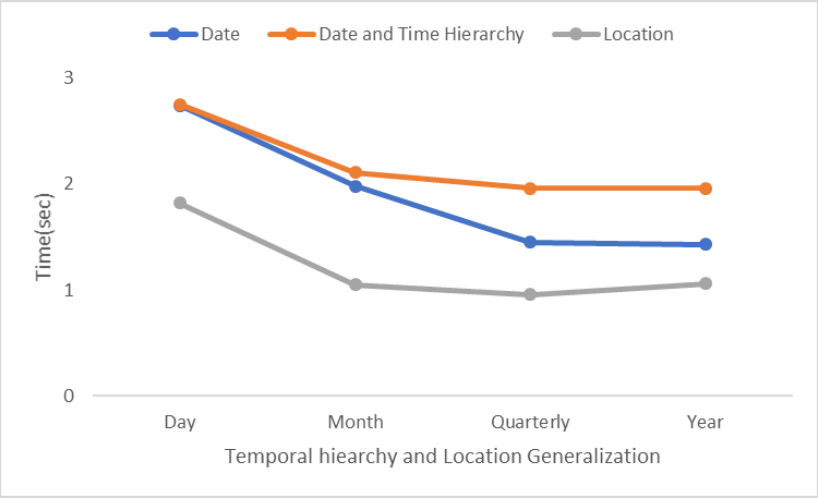


Figure 5.15: Query execution time: Buffalo data

Experiment 5.6. *Further experiments on Buffalo dataset because it has date, time and high percentage spatial coordinate (97.8 percent) appropriate for evaluation of the*

HSTPPM model.

The Buffalo dataset is pre-processed by using feature selection iteration to treat the missing values to ensure value consistency. Starting from the street, the next valid observation was used for back filling, sort the data and reload to back-fill the census block. Recall that census block is a geographic unit used by the US Census Bureau. Missing latitudes and longitudes were subsequently filled with value consistent with the specific census block. With pre-processing, 5 attributes were selected from the 18 attributes of the dataset using feature selection. Tree different representative points were explored: (a) randomly selected representative point, (b) centroid (as a weighted representative point), and (c) medroid (i.e., real data point with a minimum distance to the centroid). For each representative point, the level of spatio-temporal hierarchy was varied and the root mean square error (RMSE) of the distance between the representative point and the original data points was calculated.

Scikit-learn pre-processing RobustScaler (5,95 inter-quartile range) was utilized for standardization of the distance metrics for centroids. RobustScaler scales features using statistics that are robust to outliers. It ignores outliers, removes the median, and uses the specified inter-quartile range (IQR) for the standardization of the given set of observations. Thus, a robust-scaled feature becomes:

$$scaled\_dist = \frac{X - median(X)}{\max\{IQR(X)\} - \min\{IQR(X)\}} \tag{5.13}$$

For the random point, the resulting RMSE > 1*. Hence, MinMaxScaler was explored for the standardization of the random points distance for the purpose of calculating RMSE. MinMaxScaler is a standardization tool (from scikit-learn pre-processing) for transforming and scaling each feature in a given set of observations individually. Each scaled feature is in within the given range specified for the transformation. A range of [0, 1] was specified for the random point transformation. The transformation is in two steps as specified in Eqs. (5.14) and (5.15):*

$$dist = \frac{X - \min(X)}{\max(X) - \min(X)} \tag{5.14}$$

$$scaled\_dist = dist \times (max - min) + max \tag{5.15}$$

where min = 0 and max = 1.

Observation 5.13. *For Buffalo parking data, the result is presented in Tables 5.11 and 5.12. The spatio-temporal variation has no effect on the RMSE for the centroid and medroid approaches, however, the value of the RMSE is observed to be different at each hierarchy level for the random temporal representative point since the random selection is done at run-time.*

Table 5.11: Buffalo dataset date hierarchy RMSE

Hierarchical level	**Random**	**Centroid**	**Medroid**
Year	0.5850601	0.2756937	0.275699
Quarter	0.5302330	0.2756937	0.275699
Month	0.6000444	0.2756937	0.275699
Day	0.6110220	0.2756937	0.275699

Table 5.12: Buffalo dataset date + time hierarchy RMSE

Hierarchical level	**Random**	**Centroid**	**Medroid**
Year + time	0.5975298	0.279358914	0.279134932
Quarter + time	0.6019640	0.277100576	0.276994664
Month + time	0.6053210	0.278367390	0.278241942
Day + time	0.6105390	0.275155894	0.275086206

The result for the centroid representative point and medroid point are very close, however, it is worth noting that the centroid may not necessarily be an event location and as such may sometimes not be a good fit to represent group of observations. For instance, considering the sample parking dataset, a centroid may be in the lake, river or a location where no parking ticket has been issued. Whereas, using the medroid, we are guaranteed that a ticket has been issued at the location in the past because the point is selected from the set of observations. As shown in Section 5.8, random point has higher risk value because it is selected at run time and would essentially varies for each selection; however, similar to the medroid, note that the random point is a point within the spatial population but varies at each run time. The variation has no effect on data utility since the selection is within the observations; however, there is the

risk of presenting an actual point in some data management tasks such as preserving privacy of individuals since the randomly selected point could be traced back to the summon and hence, to individual record.

Observation 5.14. *As for* **processing time** *for Buffalo parking data, the centroid spatio-temporal grouping has smaller processing time of 0.6717 seconds while the random selection has the highest processing time of 0.85 seconds. The processing time for the medroid approach is 0.72 seconds.*

5.4.4 Evaluation Case Study 3 on the HSTPPM: COVID-19 Contact Tracing

In this section, the evaluation of the HSTPPM was done by using the **South Korean COVID-19 patient route data** which contains 5,321 trajectory points for 939 cases collected from the report materials of Korea Centers for Disease Control and Prevention (KCDC) and local governments in January-March 2020[3]. Some samples from the South Korea contact tracing are presented in Table 5.13.

Table 5.13: South Korean COVID-19 data description

Attribute	Description	Data type
PatientID	Patient numeric identifier	integer
Date	Patient activities date	date
District	patient route district	text
City	patient route city	text
Type	place visited by the patient	text
latitude	The latitude of the place visited by the patient	float
longitude	The longitude of the place visited by the patient	float

The goal for this evaluation is to minimize the utility privacy trade-off on contact tracing data such that disclosure risk does not necessary release the patient identity to the public. To do so, first the data were aggregated by *province, city and infection route* type while varying the temporal hierarchy by ***day, week and month***., then, using the medroid approach to select the spatio-temporal representative point. The procedure was repeated for aggregation by *province and infection route*.

[3]Korea Centre for Disease and Control (KCDC) 2020

Observation 5.15. *When aggregated by (province, city, and infection route), the resulting count query result is presented in Table 5.14. The aggregation by day after reclassifying all trajectories with less than minsup=3 occurrence as "others" has the lowest Pearson correlation. This is expected due to the level of granularity of presenting daily data: Lot of the occurrence that are less than three are reclassified as "others" thereby lowering the utility of the dataset. However, considering the disclosure risk, the daily aggregation is still very useful because of the time sensitive nature of the dataset. The data may become obsolete and less useful if such disclosure is delayed for instance, by a month.* The weekly aggregation provided the balance of utilities and protection with RMSE less than 0.1. *Non-disclosure of infection route with less than three count has no significant impact on the publication of the contract tracing data as shown in Table 5.15 compared with Table 5.16.*

Table 5.14: Aggregation by province, city and infection route

Infection route	**Dataset**	**Day**	**Week**	**Month**
academy	11	4	5	5
airport	120	73	95	109
bakery	24	3	3	3
bank	28	3	12	14
beauty salon	14	0	5	7
cafe	85	13	48	49
church	120	68	86	101
gas station	12	0	0	3
gym	20	6	11	12
hospital	1496	614	1238	1392
lodging	29	3	14	17
pc cafe	46	0	24	28
pharmacy	200	21	74	132
post office	15	0	4	7
public transportation	382	95	255	307
restaurant	451	98	255	326
school	49	8	22	25
store	507	157	344	397
university	14	0	3	9
others	1698	4155	2823	2378
Total	5321	5321	5321	5321

Table 5.15: RMSE with no minimum support

Aggregation level	**Day**	**Week**	**Month**
Type	0.26405	0.27500	0.29005
Province, type	0.15164	0.19226	0.20952
Province, city, type	0.06526	0.09194	0.09821

Table 5.16: RMSE minsup=3

Aggregation level	**Day**	**Week**	**Month**
Type	0.27088	0.27594	0.29027
Province, type	0.179584	0.19794	0.2114
Province, city, type	0.08631	0.09699	0.10165

Observation 5.16. *When aggregated by (province and infection route), the resulting count query is presented in Table 5.17. Disclosure at provincial level presents a lesser risk in term of the RMSE and aggregation results. The resulting aggregation by week provided the balance of utilities and privacy with root mean square error less than 0.2 and good correlation with the original dataset (0.992). The details of the RMSE and Pearson correlation for all the different variation of the temporal hierarchy are presented in Tables 5.18, 5.16 and 5.15.*

Table 5.17: Aggregation by province and infection route

Infection route	**Dataset**	**Day**	**Week**	Month
academy	11	4	5	5
airport	120	87	108	116
bakery	24	3	7	16
bank	28	3	14	21
beauty salon	14	0	5	7
cafe	85	33	61	73
church	120	74	98	113
gas station	12	0	3	6
gym	20	6	11	12
hospital	1496	1258	1456	1493
lodging	29	3	17	22
pc cafe	46	10	34	43
pharmacy	200	82	166	188
post office	15	0	5	8
public transportation	382	261	358	372
restaurant	451	273	398	438
school	49	15	32	40
store	507	346	471	493
university	14	0	4	11
others	1698	2863	2068	1844
Total	5321	5321	5321	5321

Table 5.18: Pearson correlation

Aggregation level	**Day**	**Week**	**Month**
Type	0.994935	0.996386	0.999994
Province, type	0.935215	0.991849	0.998804
Province, city, type	0.796876	0.933743	0.972514

Experiment 5.7. *This experiment focused on the data in the capital city of Seoul in South Korea. The impact of the HSTPPM using the City of Seoul having the highest route of infection was investigated.*

Observation 5.17. *Observations from classifying infection route with less than three*

trajectories as "others" route include:

1. *At the* daily hierarchy level*, non-disclosure of some route such as bakery, bank, beauty cafe, gas station, gym, PC cafe (i.e., internet cafe), post office, salon, and school were noticeable.*

2. *At the* weekly temporal hierarchy level*, the major observation was the non-disclosure of some route such as beauty salon, gym and post office infection route.*

3. *At the* monthly temporal hierarchy level*, three infection routes—namely, beauty salon, gym and post office—were not disclosed.*

The resulting count query is presented in Table 5.19. However, non-disclosure does not necessarily translate to no contact tracing because the city may still trace the contact on those infection routes without disclosing the routes.

Table 5.19: Aggregation effect on Seoul data

Infection route	Dataset	Day	Week	Month
airport	18	13	18	18
bakery	6			4
bank	8		5	6
beauty salon	3			
cafe	30	10	24	30
church	37	17	31	37
gas station	3		3	3
gym	3			
hospital	603	582	601	603
lodging	8	3	6	8
PC cafe	19		14	19
pharmacy	53	30	51	53
post office	3			
public transportation	192	166	191	192
restaurant	150	120	146	150
school	5			3
store	146	117	146	146
others	424	653	475	439
Total	1711	1711	1711	1711

5.4.5 Evaluation on the DP-HSTPPM for Contact Tracing

Experiment 5.8. *In this experiment, HSTPPM was extended by incorporating differential privacy (DP) which has privacy guaranteed and resulted in DP-based HSTPPM called* **DP-HSTPPM**. *To evaluate its practicality for contact tracing, The following steps were performed to the HSTPPM.*

- *added noise to individual point varying the noise parameter,*
- *aggregated the data by province, city and infection route type,*
- *varied the temporal hierarchy by day, week and month; and*
- *calculate medroid for each temporal hierarchy, as well as the root mean square error (RMSE).*

Observation 5.18. *It was observed that adding DP (reasonably controlled noise parameter) to the HSTPPM has no significant impact on the utility of the dataset as summarized in Tables 5.20, 5.21, 5.22 and 5.23. The RMSE is 0.079 where ϵ =0.01 compared with 0.065 recorded for the the resulting DP-HSTPPM.*

Table 5.20: Impact of differential privacy on daily temporal hierarchy

Epsilon	RMSE	Distance (km)
0.01	0.07857	0.05214
0.02	0.14112	0.09733
0.03	0.16238	0.11423
0.04	0.18574	0.13053
0.05	0.17393	0.12311
0.06	0.18163	0.12681
0.07	0.18194	0.12674
0.08	0.09797	0.06758
0.09	0.17593	0.12270
0.10	0.17048	0.11955

Table 5.21: Impact of differential privacy on weekly temporal hierarchy

Epsilon	RMSE	Distance (km)
0.01	0.10939	0.07120
0.02	0.16665	0.11622
0.03	0.17317	0.11911
0.04	0.17662	0.12295
0.05	0.16529	0.11398
0.06	0.13857	0.09659
0.07	0.15736	0.10568
0.08	0.13725	0.09488
0.09	0.17753	0.12191
0.10	0.19154	0.13617

Table 5.22: Impact of differential privacy on monthly temporal hierarchy

Epsilon	RMSE	Distance (km)
0.01	0.12454	0.07660
0.02	0.19719	0.13487
0.03	0.13838	0.09506
0.04	0.13990	0.09542
0.05	0.19291	0.13086
0.06	0.16877	0.11256
0.07	0.16352	0.10964
0.08	0.16352	0.11166
0.09	0.19449	0.13138
0.10	0.12017	0.08023

Table 5.23: RMSE summary for HSTPPM and DPHSTPPM

Aggregation level: Province, city, type; $\epsilon = 0.01$		
Hierarchy	HSTPPM	DP-HSTPPM
Day	0.06526	0.07860
Week	0.09194	0.10939
Month	0.09821	0.12455

5.4.6 Evaluation on the GAN-HSTPPM for Contact Tracing

Experiment 5.9. *The HSTPPM was extended by incorporating generative adversarial network (GAN) and resulted in GAN-based HSTPPM called* **GAN-HSTPPM**. *To evaluate its practicality for contact tracing, The Nash equilibrium and effect of using the machine learning generative adversarial perturbation method for contact tracing were investigated using series of experiments. The setup and architecture are summarized in Table 5.24. The training steps:*

- *varied the training data percentage (20%, 40%, 60% and 80%),*
- *trained the generator by taking a batch of 32 datasets for each epoch. The*

same size was generated through the generator. The input to the generator was random numbers from standard normal distribution with a mean μ=0 and variance=1,

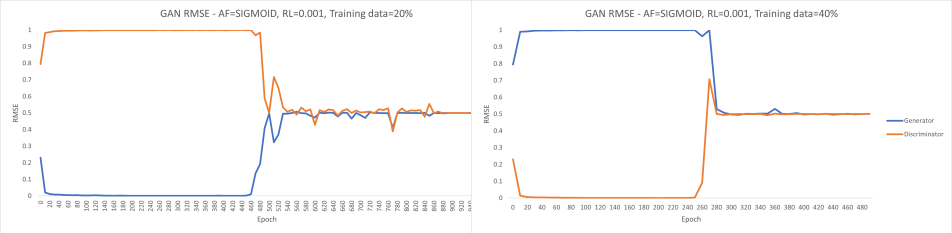
- *trained the discriminator by combining real data and generated data for each batch size; and*
- *calculated the RMSE for the generator and the discriminator.*

Further investigation was done by applying different activation functions—such as rectified linear unit function (ReLU), sigmoid, and/or tanh.

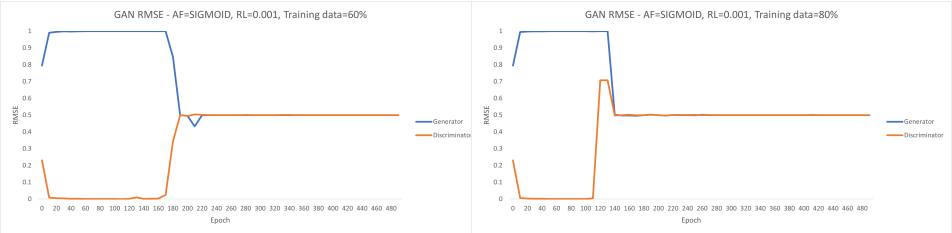
Table 5.24: GAN architecture for private contact tracing

epoch=500, learning rate=0.001, optimizer=Adam Activation function = ReLU/sigmoid/tanh loss function=MSE, batch size=32		
Model	No of Hidden Layers	No of Neurons
Trajectory discriminator	3	256, 128, 64
Trajectory generator	2	32, 16

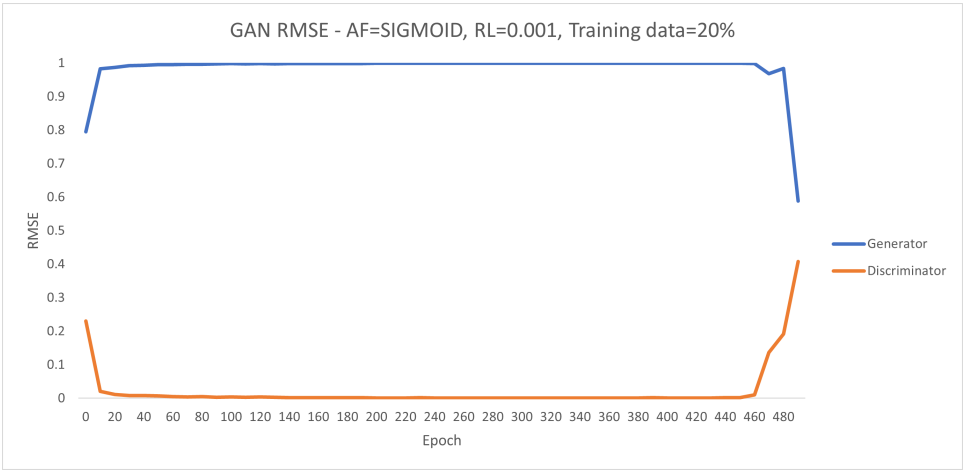
Observation 5.19. *The difficulty in attaining Nash equilibrium was observed using 20% of dataset for training, 500 training loop (i.e. epoch=500) with a wide gap between the generator and discriminator RMSE as shown in Figures 5.16e and 5.17e for both sigmoid and tanh activation functions with no equilibrium in sight. Hence, the training loop for the 20% dataset training was increased to 1000. A near equilibrium around epoch=504 for sigmoid activation function where the generator RMSE=0.50036 and the discriminator RMSE =0.49738 with a difference of 0.003 was observed as shown in Figure 5.16a. The difficulty is more severe for tanh activation function as shown in Figure 5.17a, the first near equilibrium was observed when epoch=976 where the generator RMSE=0.49506 and the discriminator RMSE =0.49241 with a difference of 0.003.*



(a) sigmoid 20 (b) sigmoid 40

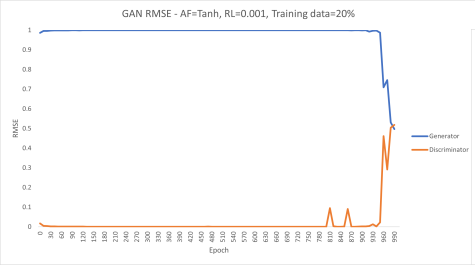


(c) sigmoid 60 (d) sigmoid 80

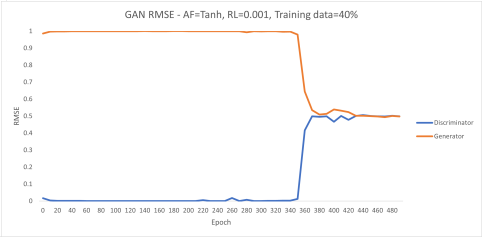


(e) sigmoid Nash difficulty

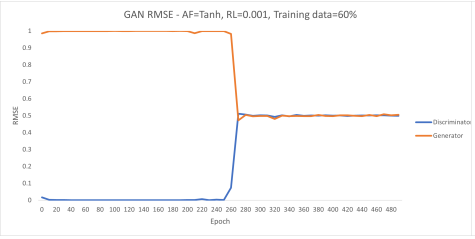
Figure 5.16: GAN RMSE sigmoid activation function



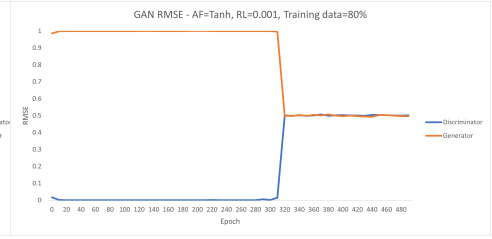
(a) tanh 20



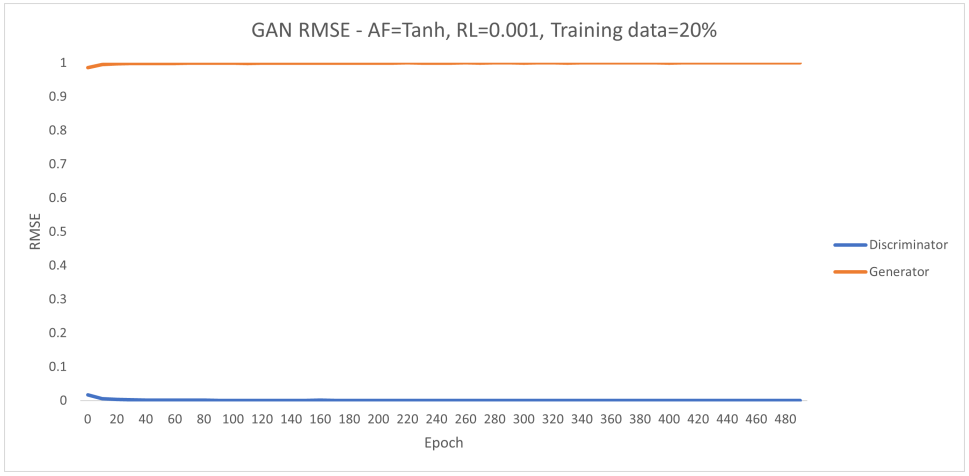
(b) tanh 40



(c) tanh 60



(d) tanh 80



(e) tanh Nash difficulty

Figure 5.17: GAN RMSE tanh activation function

Observation 5.20. *Regarding the* impact of activation function, *early good result is observed for ReLU activation function:*

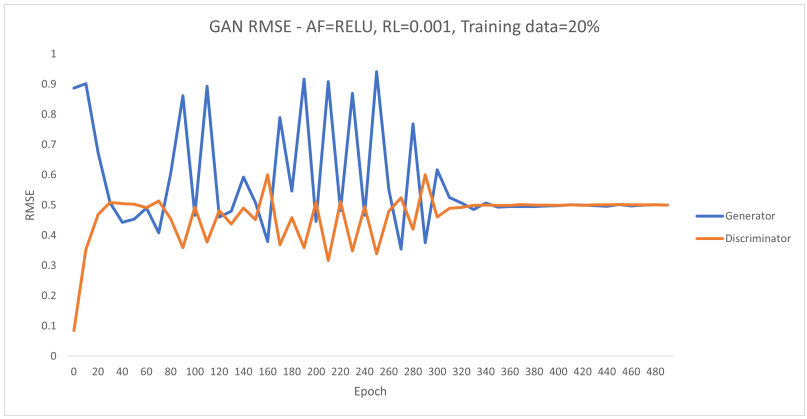
- *20% dataset: epoch=60 with RMSE difference of 0.0007 and epoch=480 with RMSE difference of 0.0001.*
- *40% dataset: epoch=60 with RMSE difference of 0.00001*

- *60% dataset: epoch=80 with RMSE difference of 0.0002 and epoch=390 with RMSE difference of 0.00001.*
- *80% dataset: epoch=150 with RMSE difference of 0.00001.*

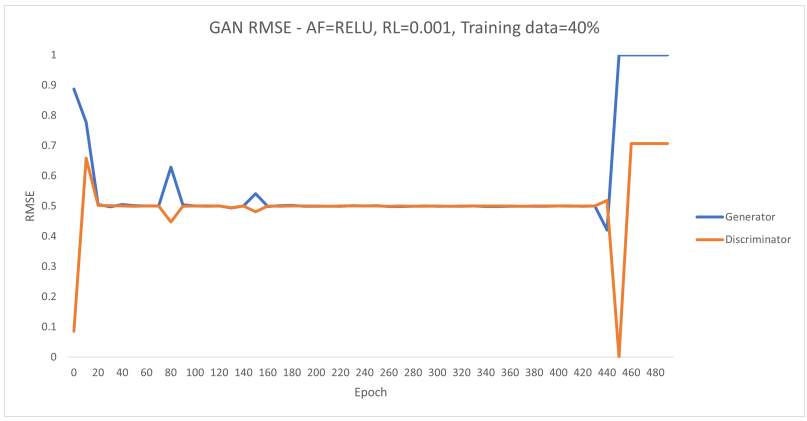
The summary for the generator RMSE and discriminator RMSE is presented in Table 5.25, as well as Figures 5.16, 5.17 and 5.18. Tanh activation function has higher RMSE difference, increasing the training loop (epoch), adding more hidden layer and/or using high volume dataset may improve the performance. Further investigation was done with additional hidden layer and recorded an RMSE difference of 0.0001 at epoch=970 for 80% dataset training. However, the outcome is not significantly different from the previous result.

Table 5.25: Impact of activation function on GAN

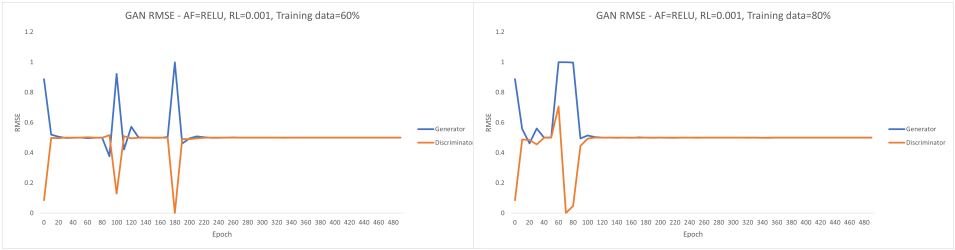
	ReLU		Sigmoid		Tanh	
Training%	Epoch	RMSE dif	Epoch	RMSE dif	Epoch	RMSE dif
20%	480	0.00010	920	0.00010	994	0.0020
40%	60	0.00001	490	0.00001	490	0.0008
60%	390	0.00001	420	0.00001	410	0.0005
80%	150	0.00001	300	0.00001	460	0.0002
RMSE dif = Generator, Discriminator RMSE difference						



(a) ReLU 20



(b) ReLU 40



(c) ReLU 60 (d) ReLU 80

Figure 5.18: GAN RMSE ReLU activation function

5.4.7 Comparisons among HSTPPM, DP-HSTPPM and GAN-HSTPPM

The summary of the HSTPPM, DP-HSTPPM and GAN-HSTPPM is presented

in Table 5.26. The result for ReLU activation function with 80% dataset of training is used as the basis for comparison. The HSTPPM provides good balance of utility and privacy by regrouping and merging cases where a group has less than the minimum support threshold rather than adding noise (DP-HSTPPM) or generating artificial privacy preserving data (GAN-hSTPPM).The comparison shows that the differential privacy which has privacy guaranteed with well controlled epsilon ($\epsilon = 0.001$ for this experiment) can provide good balance of utility. Generative adversarial network looks promising, however there is still more work to be done to resolve the issue of difficulty in achieving Nash equilibrium.

Table 5.26: RMSE summary for HSTPPM, DP-HSTPPM & GAN-HSTPPM

Aggregation level: Province, city, type			
Hierarchy	HSTPPM	DP-HSTPPM	GAN-HSTPPM
Day	0.06526	0.0786	0.50001

5.5 Summary

After designing and implementing a hierarchical spatio-temporal model (HSTM) in an open data lake architecture with a metadata collection framework in the previous chapter, this chapter explores how to preserve privacy in the HSTM designing and implementing a hierarchical spatio-temporal privacy preserving model (HSTPPM) by non-trivially incorporating privacy preservation into the HSTM. Moreover, further extension of the HSTPPM was done to tasks like (a) privacy-preserving publishing and (b) COVID-19 contact tracing. Furthermore, the HSTPPM was extended to (a) DP-HSTPPM and (b) GAN-HSTPPM by incorporating different privacy-preserving techniques like differential privacy (DP) and generative adversarial network (GAN), respectively.

To elaborate, in this chapter, the hierarchical spatio-temporal privacy preserving model (HSTPPM) comprising of (a) temporal hierarchy (with extension to time generalization), (b) location hierarchy and (c) temporal representative points was

presented.

Next, the following big data algorithms integrating the components of the hierarchical model were presented :

1. The hierarchical temporal location model: The application of the model was demonstrated with example of how individual privacy in big dataset could be preserved in the hierarchy of time and location.

2. The hierarchical spatio-temporal model: The hierarchical temporal location was improved to include situation where the actual location is provided in form of spatial dataset by adding computation of temporal representative point to the hierarchy.

Next, several experiments were performed to evaluate the hierarchical spatio-temporal privacy preserving models using real life big dataset. The evaluation includes:

1. Privacy preserving bylaw enforcement data publishing, compression and visualizer. The privacy preserving model was applied to bylaw enforcement parking dataset publishing aggregating the dataset to different level of the hierarchy. The approach is flexible such that user can select the level of privacy appropriate for the situation. The model is also applicable to big data compression based on the level of hierarchy selected, the higher the hierarchy, the lower the processing time and disclosure risk.

2. Privacy preserving contact tracing of COVID-19. The major observation for this experiment is that weekly publishing of COVID-19 contact tracing data provides the balance of utility and privacy with minimum disclosure risk. The solution preserves the privacy of individuals by (a) building a temporal hierarchy, (b) grouping similar data to preserve the actual timestamp and locations, (c) representing each group by their spatio-temporal representative points.

3. The hierarchical spatio-temporal privacy preserving model (HSTPPM) was compared with differential privacy hierarchical spatio-temporal privacy preserving model (DP-HSTPPM) and generative adversarial network (GAN-HSTPPM)

model. The empirical evidence showed that there is no significant difference between HSTPPM and adding well controlled Laplace Differential privacy noise to the model. The DP-HSTPPM has privacy guaranteed. However, GAN model has higher RMSE which may improve privacy but distort the dataset beyond its usefulness.

Chapter 6

Co-occurrence Pattern Mining and Visualization with Hierarchical Spatio-temporal Privacy Preserving Model

In addition to privacy-preserving publishing and contact tracing, the HSTPPM can also be incorporated into co-occurrence pattern mining. The key contributions of this chapter include the design and implementation of (a) the co-occurrence pattern mining (i.e., data analytics) with the HSTPPM and (b) visual analytics with the HSTPPM.

6.1 Pattern Mining with the HSTPPM

In this section, three co-occurrence patterns that could benefit from the hierarchical spatio-temporal model are identified as follows:

1. Temporal co-occurrence of events: Events $(e_1), (e_2), (e_3), ..., (e_n)$ occurring at the same temporal hierarchy level t_1 in different location $(l_1), (l_2), (l_3), ..., (l_n)$ such that we have set of attributes $(e_1, t_1, l_1), (e_1, t_1, l_2), (e_1, t_1, l_3), ..., (e_1, t_1, l_n)$

and $(e_1), (e_2), (e_3), ..., (e_n)$. The event may be the same type of event such as COVID-19 occurring all over the world between 2020-2022 or multi-events. The goal is to leverage spatio-temporal hierarchy to analyze and understand the temporal co-occurrence of events across different locations.

2. Spatial co-occurrence of events: Events occurring at the same spatial hierarchy level (l_1) but at different temporal hierarchy such that we have (e_1, t_1, l_1), $(e_2, t_2, l_1), (e_3, t_3, l_1), ..., (e_n, t_n, l_1)$. The goal is to leverage the hierarchical spatio-temporal privacy preserving model to analyze and understand the spatial co-occurrence of events across different time intervals.

3. Spatio-temporal co-occurrence: refers to events occurring at different temporal hierarchy and/or in different locations such that we have $(e_1, t_1, l_1), (e_2, t_2, l_2)$, $(e_3, t_3, l_3), ..., (e_n, t_n, l_n)$. For instance, considering Tables 1.2 and 1.3, one can infer that the parking ticket event co-occur with the hospital visit event in the city of Winnipeg on 12 March 2018.

Step 1: Compute the co-occurrence matrix by modifying the hierarchical spatio-temporal model (HSTM) as follows:

1. Select the temporal hierarchy level,

2. For each row in the dataset, set the row to 1 if an event exists and 0 otherwise.

Step 2: Compute the vertical count of ones for single event and dot product for multiple events co-occurrence pattern. Interesting co-occurrence pattern mining becomes trivial with the co-occurrence matrix.

Example 6.1. *The temporal co-occurrence of COVID-19 for all provinces and territories in Canada in the first ten days of the year 2021 can be represented using co-occurrence bit matrix shown in Table 6.1. For single province, the action is the simple count the vertical number of ones. Finding pattern in multiple provinces requires counting the dot product of their bit matrix. for instance, to find the common pattern for the Prairie provinces from Table 6.1, involves computing the dot product of*

Algorithm 6.1 Temporal co-occurrence model

```
Input: Spatio-temporal Dataset D
Output: Temporal co-occurrence matrix D'
Date ← create temporal hierarchy  ▷ create location hierarchy for spatial model
for selected temporal hierarchy do
    for each Location, l_i do                    ▷ temporal for location hierarchy
        if value > minimumsupport then           ▷ varies minimum support
            row ← 1
        elserow ← 0
    Spatio-temporal Pattern ← Count(row ← 1)     ▷ single location
    Spatio-temporal Pattern ← Count((l_1) · (l_2) · (l_3) · (..l_n))   ▷ multiple
locations
```

bits for Alberta, Manitoba and Saskatchewan as shown in Figure 6.1. The cross product showing the co-occurrence pattern is presented in Table 6.2. The spatio-temporal co-occurrence pattern observations include:

1. *Alberta, British Columbia, New Brunswick, Ontario, Quebec, and Saskatchewan reported at least a case per day.*

2. *Manitoba and Nova Scotia reported cases for 9 out of the 10 days.*

3. *Yukon has cases reported in 5 out of the 10 days*

4. *Newfoundland and Labrador, and Prince Edward Island have cases reported in 3 out of the 10 days.*

5. *Further pruning of the matrix by increasing the minimum support to 500 as presented in Table 6.3 shows that only 2 (Ontario and Quebec) out of 13 provinces in Canada recorded 500 or more cases per day while Alberta recorded over 500 cases per day in 9 days and British Columbia has 8 days record of over 500 cases per day in the first 10 days of 2021.*

Table 6.1: Spatio-temporal co-occurrence bit matrix for Canadian COVID-19

Date	AB	BC	MB	NB	NL	NS	ON	PE	QC	SK	YK
2021-01-01	1	1	0	1	0	1	1	0	1	1	1
2021-01-02	1	1	1	1	0	1	1	0	1	1	0
2021-01-03	1	1	1	1	0	1	1	0	1	1	0
2021-01-04	1	1	1	1	1	1	1	0	1	1	1
2021-01-05	1	1	1	1	1	1	1	1	1	1	0
2021-01-06	1	1	1	1	0	1	1	1	1	1	1
2021-01-07	1	1	1	1	0	1	1	1	1	1	1
2021-01-08	1	1	1	1	0	1	1	0	1	1	1
2021-01-09	1	1	1	1	0	1	1	0	1	1	0
2021-01-10	1	1	1	1	1	0	1	0	1	1	0

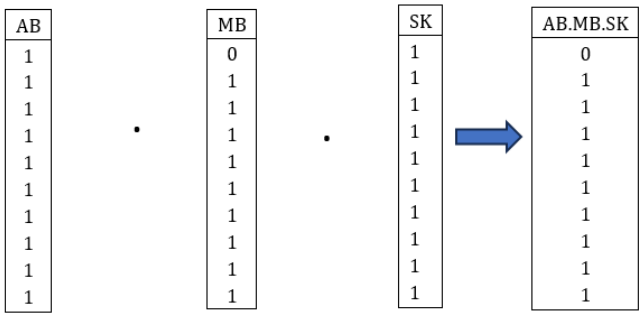


Figure 6.1: Prairie provinces co-occurrence pattern

Table 6.2: COVID-19 temporal co-occurrence pattern in Canadian provinces

Province	AB	BC	MB	NB	NL	NS	ON	PE	QC	SK	YK
AB	**10**	10	9	10	3	9	10	3	10	10	5
BC	10	**10**	9	10	3	9	10	3	10	10	5
MB	9	9	**9**	9	3	9	9	3	9	9	5
NB	10	10	9	**10**	3	9	10	3	10	10	5
NL	3	3	3	3	**3**	3	3	3	3	3	3
NS	9	9	9	9	3	**9**	9	3	9	9	5
ON	10	10	9	10	3	9	**10**	3	10	10	5
PE	3	3	3	3	3	3	3	**3**	3	3	3
QC	10	10	9	10	3	9	10	3	**10**	10	5
SK	10	10	9	10	3	9	10	3	10	**10**	5
YK	5	5	5	5	3	5	5	3	5	5	**5**

Table 6.3: COVID-19 temporal co-occurrence matrix, minsup=500

Date	AB	BC	ON	QC
2021-01-01	1	1	1	1
2021-01-02	1	1	1	1
2021-01-03	0	0	1	1
2021-01-04	1	1	1	1
2021-01-05	1	0	1	1
2021-01-06	1	1	1	1
2021-01-07	1	1	1	1
2021-01-08	1	1	1	1
2021-01-09	1	1	1	1
2021-01-10	1	1	1	1

6.2 Pattern Visualization of the HSTPPM Data

As previously pointed out, the HSTPPM is applicable to different areas of big data management tasks, thus the model is applied to privacy-preserving visualization of spatio-temporal data in this section.

- Use HSTPPM to visualize and find common pattern (e.g., between by law enforcement in two cities like (a) the City of Buffalo, NY, USA and (b) the City of Winnipeg, MB, Canada).
- Visualize the different spatial pattern in the three temporal representative point approaches: Centroid, random, and medroid points.

Specifically, the common pattern task was done using the year, month and day hierarchy levels. The tools utilized for the tasks include GeoPandas and Matplotlib for temporal visualization and folium plugins for spatial visualization. GeoPandas is an open-source geo-spatial data processing library in Python, whereas Matplotlib is a plotting library for the Python and its numerical mathematics extension NumPy.

6.3 Evaluation

6.3.1 Evaluation on Data Analytics/Pattern Mining from the HSTPPM Data

In this section, the real-life application of the HSTPPM to co-occurrence pattern mining is demonstrated with empirical analysis using Canada COVID-19 dataset:

- **Canada COVID-19 Data**[1], which is the Canada COVID-19 containing 9,957 daily records of cases, death and recovered from 31 January 2020 to 30 December 2021. Detail description of the data is presented in Table 6.4.

[1]https://health-infobase.canada.ca/covid-19/epidemiological-summary-covid-19-cases.html

Table 6.4: Canadian COVID-19 data description

Attribute	**Description**	**Data type**
pruid	Provincial ID	integer
prname	Province name (English)	text
prnameFR	Province name (French)	text
date	Report date	date
numconf	Number of confirmed cases	integer
numprob	Number of probable cases	integer
numdeaths	Number of deaths	integer
numtotal	Total number of cases	integer
numtested	Number of individuals tested	integer
numrecover	Number of recovered cases	integer
percentrecover	Percentage of recovered cases	float
ratetested	Testing rate per one million population	float
numtoday	Number of new cases since last update	integer
percentoday	Percent change since last update	float
ratetotal	Case rate per one hundred thousand population	float

Experiment 6.1. *In this experiment, the aim is to apply the designed spatio-temporal co-occurrence algorithm to mine spatio-temporal co-occurrence pattern from Canada COVID-19 dataset. Specifically, the experiment actions include:*

- *varied the minimum support threshold which is the minimum number of cases that must be recorded to be included in the co-occurrence matrix and measured the processing time.*
- *varied the hierarchical level to DAY, WEEK and MONTH. There is a total of 675 days in the dataset spanning 24 months and 101 weeks. The resulting co-occurrence pattern across Canada provinces was recorded for further analysis.*

Observation 6.1. *For DAY Hierarchical spatio-temporal co-occurrence pattern, the processing time remain relatively constant across the minimum support as shown in Figure 6.2 ranging from 0.847 for minimum support of 10 and 0.314 for the minimum support of 500 cases.*

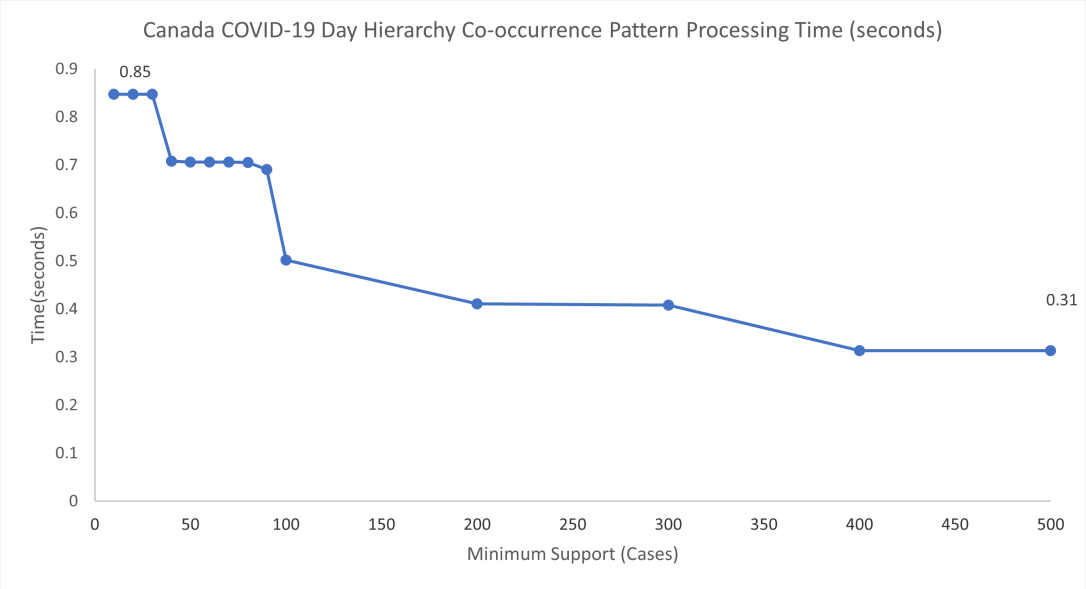


Figure 6.2: Canadian COVID-19 daily hierarchy co-occurrence pattern processing time

The temporal Co-Occurrence for minimum support of 100,200,300,400 and 500 are presented in Tables 6.5, 6.6, 6.7, 6.8 and 6.9:

1. ***Day hierarchy minimum support 100**:*

   - *Ten out of the thirteen provinces in Canada recorded over 100 cases for at least 3 days each.*
   - *Northwest Territories Nunavut and Yukon have no record of over 100 cases for any day throughout the pandemic period.*
   - *Ontario has the highest record of over 100 cases per day.*
   - *Quebec over 200 cases co-occurred 90% with Ontario cases.*

- *Manitoba, Nova Scotia, New Brunswick, Newfoundland and Labrador and Saskatchewan cases co-occurred with Ontario cases 100 % of the day hierarchical level.*
- *Manitoba over 100 cases (255 days) co-occurred with Saskatchewan cases 72% (183 days) of the day hierarchical level.*

Table 6.5: Canadian COVID-19 temporal co-occurrence pattern, minsup=100 cases

DAY Co-occurrence Pattern where minsup = 100										
Province	AB	BC	MB	NB	NL	NS	ON	PE	QC	SK
AB	**478**	427	250	28	4	37	471	3	460	308
BC	427	**432**	247	28	4	37	428	3	421	307
MB	250	247	**255**	27	4	37	255	3	250	183
NB	28	28	27	**28**	4	20	28	3	25	15
NL	4	4	4	4	**4**	4	4	3	4	4
NS	37	37	37	20	4	**37**	37	3	34	25
ON	471	428	255	28	4	37	**624**	3	566	309
PE	3	3	3	3	3	3	3	**3**	3	3
QC	460	421	250	25	4	34	566	3	**582**	303
SK	308	307	183	15	4	25	309	3	303	**309**

2. ***Day hierarchy minimum support 200 cases**:*

- *Nine out of the thirteen provinces in Canada recorded over 200 cases for at least 3 days each.*
- *In addition to Northwest Territories Nunavut and Yukon, Prince Edward Island has ho record of over 200 cases.*
- *Manitoba over 200 cases co-occurred with Saskatchewan only 41% of the day hierarchical level frequency.*

Table 6.6: Canadian COVID-19 temporal co-occurrence pattern; minsup=200 cases

DAY Co-occurrence Pattern where minsup = 200									
Province	AB	BC	MB	NB	NL	NS	ON	QC	SK
AB	**411**	31	124	7	3	16	402	392	174
BC	31	**384**	118	7	3	16	379	371	174
MB	124	118	**128**	7	3	14	128	119	53
NB	7	7	7	**7**	2	7	7	5	4
NL	3	3	3	2	**3**	3	3	3	2
NS	16	16	14	7	3	**16**	16	13	5
ON	402	379	128	7	3	16	**530**	484	174
QC	392	371	119	5	3	13	484	**495**	171
SK	174	174	53	4	2	5	174	171	**174**

3. ***Day hierarchy minimum support 300 cases**:*

   - *Nine out of the thirteen provinces in Canada recorded over 300 cases for at least 3 days each.*
   - *Manitoba and Saskatchewan recorded over 300 cases for 69 days each, however only 10% co-occurred on the same day.*

Table 6.7: Canadian COVID-19 temporal co-occurrence pattern; minsup=300 cases

DAY Co-occurrence Pattern where minsup = 300									
Province	AB	BC	MB	NB	NL	NS	ON	QC	SK
AB	**359**	330	67	4	3	14	352	344	69
BC	330	**358**	63	4	3	14	355	348	69
MB	67	63	**69**	4	3	10	69	64	7
NB	4	4	4	**4**	3	4	4	3	1
NL	3	3	3	3	**3**	3	3	3	3
NS	14	14	10	4	3	**14**	14	11	1
ON	352	355	69	4	3	14	**492**	452	69
QC	344	348	64	3	3	11	452	**467**	69
SK	69	69	7	1	1	1	69	69	**69**

4. ***Day hierarchy minimum support 400 cases**:*

- *Eight out of the thirteen provinces in Canada recorded over 400 cases for at least 2 days each.*
- *Quebec has the highest number of days record for over 400 cases*
- *Manitoba and Saskatchewan cases over 400 cases co-occurred 3% on the day hierarchical level.*

Table 6.8: Canadian COVID-19 temporal co-occurrence pattern; minsup=400 cases

DAY Co-occurrence Pattern where minsup = 400								
Province	AB	BC	MB	NB	NS	ON	QC	SK
AB	**296**	264	38	2	13	280	279	32
BC	264	**312**	33	2	13	293	294	32
MB	38	33	**38**	2	9	38	35	1
NB	2	2	2	**2**	2	2	2	1
NS	13	13	9	2	**13**	13	10	1
ON	280	293	38	2	13	**435**	403	32
QC	279	294	35	2	10	403	**446**	32
SK	32	32	1	1	1	32	32	**32**

5. ***Day hierarchy minimum support 500 cases**:*

- *Eight out of the thirteen provinces in Canada recorded over 500 cases for at least 1 day each.*
- *Quebec has the highest number of days for over 500 cases and 82% co-occurred with Ontario over 500 cases.*
- *New Brunswick has only one day record of over 500 cases*

Table 6.9: Canadian COVID-19 temporal co-occurrence pattern; minsup=500 cases

DAY Co-occurrence Pattern where minsup = 500								
Province	AB	BC	MB	NB	NS	ON	QC	SK
AB	**248**	209	15	1	10	220	215	11
BC	209	**238**	12	1	10	212	211	10
MB	15	12	**16**	1	8	16	13	1
NB	1	1	1	**1**	1	1	1	1
NS	10	10	8	1	**10**	10	7	1
ON	220	212	16	1	10	**371**	335	11
QC	215	211	13	1	7	335	**408**	11
SK	11	10	1	1	1	11	11	**11**

Observation 6.2. *For **WEEK Hierarchical spatio-temporal co-occurrence pattern**, the processing time is presented in Figure 6.3 ranging from 0.486 for the minimum support of 500 cases to 0.247 for minimum support of 2000 cases.*

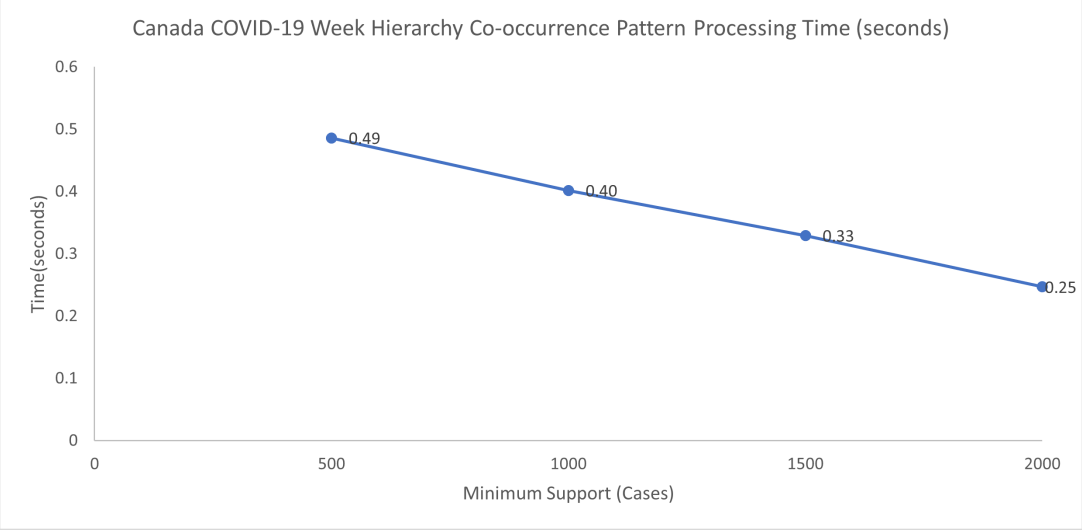


Figure 6.3: Canadian COVID-19 weekly hierarchy co-occurrence pattern processing time

The week hierarchy co-occurrence pattern for Canada COVID-19 dataset with the minimum support of 500,1000,1500 and 2000 are presented in Tables 6.10, 6.11, 6.12 and 6.13:

1. ***Week hierarchy minimum support 500 cases***

   - *Ten out of the thirteen provinces in Canada recorded over 500 cases for at least one week each.*
   - *Northwest Territories Nunavut and Yukon have ho record of over 500 cases per week.*
   - *Ontario has the highest record of over 500 cases per week for 93 weeks out of the 101 weeks pandemic period in the dataset.*
   - *All other provinces (namely, Alberta, British Columbia, Manitoba, New Brunswick, Newfoundland and Labrador, Nova Scotia, Prince Edward Island, Quebec and Saskatchewan) have 100% co-occurrence with Ontario.*
   - *Newfoundland and Labrador and Prince Edward Island recorded over 500 cases for only one week.*

Table 6.10: Canadian COVID-19 weekly temporal co-occurrence pattern; minsup=500 cases

WEEK Co-occurrence Pattern where minsup = 500										
Province	AB	BC	MB	NB	NL	NS	ON	PE	QC	SK
AB	**77**	68	47	9	1	8	77	1	76	50
BC	68	**68**	47	9	1	8	68	1	67	50
MB	47	47	**48**	9	1	8	48	1	47	40
NB	9	9	9	**9**	1	4	9	1	9	6
NL	1	1	1	1	**1**	1	1	1	1	1
NS	8	8	8	4	1	**8**	8	1	8	6
ON	77	68	48	9	1	8	**93**	1	89	50
PE	1	1	1	1	1	1	1	**1**	1	1
QC	76	67	47	9	1	8	89	1	**89**	50
SK	50	50	40	6	1	6	50	1	50	**50**

2. ***Week hierarchy minimum support 1000 cases***

   - *Nine out of the thirteen provinces in Canada recorded over 1000 cases for at least one week each.*

- *Ontario has the highest record of over 1000 cases per week for 84 weeks out of the 101 weeks pandemic period in the dataset.*
- *Four provinces (Northwest Territories Nunavut, Prince Edward Island and Yukon) have no record of over 1000 cases per week.*

Table 6.11: Canadian COVID-19 weekly temporal co-occurrence pattern; minsup=1000 cases

WEEK Co-occurrence Pattern where minsup = 1000									
Province	AB	BC	MB	NB	NL	NS	ON	QC	SK
AB	**62**	57	27	2	1	4	62	61	42
BC	57	**57**	26	2	1	4	57	56	42
MB	27	26	**27**	2	1	4	27	27	18
NB	2	2	2	**2**	1	2	2	2	2
NL	1	1	1	1	**1**	1	1	1	1
NS	4	4	4	2	1	**4**	4	4	3
ON	62	57	27	2	1	4	**84**	74	42
QC	61	56	27	2	1	4	74	**77**	42
SK	42	42	18	2	1	3	42	42	**42**

3. ***Week hierarchy minimum support 1500 cases***

- *Eight out of the thirteen provinces in Canada recorded over 1500 cases for at least two weeks each.*
- *Five provinces (Nunavut, Prince Edward Island, Newfoundland and Labrador, Northwest Territories and Yukon) have no record of over 1500 cases per week.*
- *New Brunswick has 2 weeks,Nova Scotia has 3 weeks and Saskatchewan has 23 weeks, however, the 3 provinces have no co-occurrence pattern for any week with over 1500 cases.*

Table 6.12: Canadian COVID-19 weekly temporal co-occurrence pattern; minsup=1500 cases

WEEK Co-occurrence Pattern where minsup > 1500								
Province	AB	BC	MB	NB	NS	ON	QC	SK
AB	58	53	17	2	3	58	58	23
BC	53	53	16	2	3	53	53	23
MB	17	16	18	2	3	18	17	6
NB	2	2	2	2	2	2	2	0
NS	3	3	3	2	3	3	3	0
ON	58	53	18	2	3	75	70	23
QC	58	53	17	2	3	70	71	23
SK	23	23	6	0	0	23	23	23

4. ***Weekly minimum support 2000 cases***

- *Seven out of the thirteen provinces in Canada recorded over 2000 cases for at least three weeks each.*
- *Six provinces (New Brunswick, Nunavut, Prince Edward Island, Newfoundland and Labrador, Northwest Territories and Yukon) have no record of over 2,000 cases per week.*
- *Manitoba has 12 weeks, Nova Scotia has 3 weeks and Saskatchewan has 9 weeks, however, the 3 provinces have no co-occurrence pattern for any week with over 2000 cases.*

Table 6.13: Canadian COVID-19 weekly hierarchy temporal co-occurrence pattern; minsup > 2k cases

WEEK Co-occurrence Pattern where minsup = 2000							
Province	AB	BC	MB	NS	ON	QC	SK
AB	54	53	12	3	53	53	9
BC	53	53	12	3	52	52	9
MB	12	12	12	2	12	12	0
NS	3	3	2	3	3	3	0
ON	53	52	12	3	72	67	9
QC	53	52	12	3	67	68	9
SK	9	9	0	0	9	9	9

Observation 6.3. *For **MONTH Hierarchical spatio-temporal co-occurrence pattern**, the processing time is presented in Figure 6.4 having the lowest processing time of 0.17 seconds for the minimum support of over 10,000 cases per month and the highest processing time of 0.31 seconds for 2000 cases per month.*

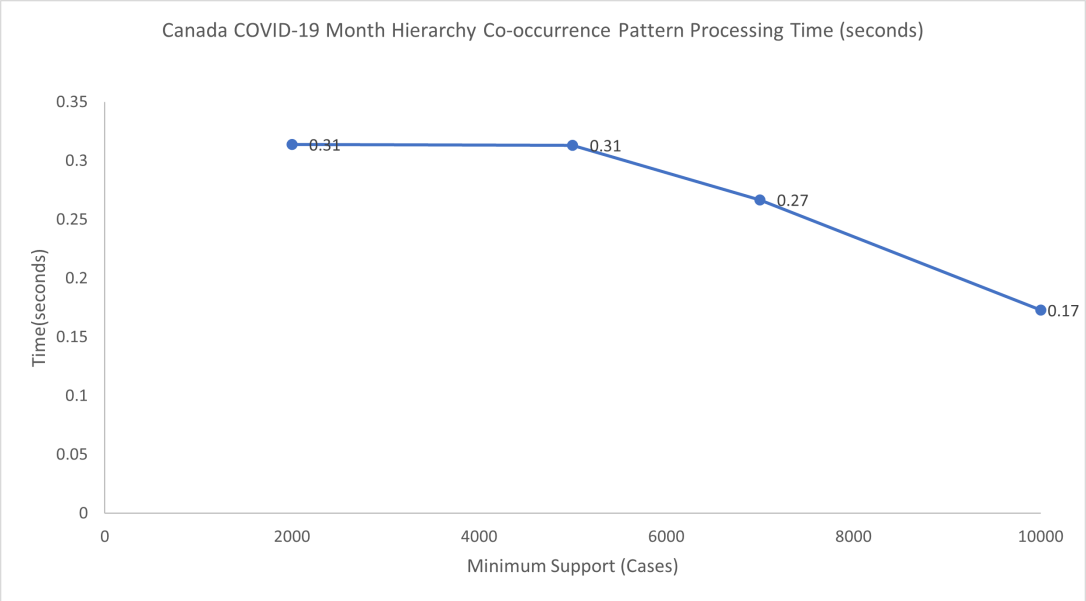


Figure 6.4: Canadian COVID-19 monthly hierarchy co-occurrence pattern processing time

The month hierarchy co-occurrence pattern for Canada COVID-19 dataset with

the minimum support of 2000, 5000, 7000 and 10000 are presented in Tables 6.14, 6.15, 6.16 and 6.17:

1. ***Month hierarchy minimum support 2000 cases***

   - *Eight out of the thirteen provinces in Canada recorded over 2000 cases for at least two months each.*
   - *Five provinces (Nunavut, Prince Edward Island, Newfoundland and Labrador, Northwest Territories and Yukon) have no record of over 2000 cases per month.*
   - *Quebec has the highest record of over 2000 cases per month for 22 months out of the 24 months pandemic period in the dataset.*

Table 6.14: Canadian COVID-19 monthly hierarchy temporal co-occurrence pattern; minsup=2000 cases

MONTH Co-occurrence Pattern where minsup > 2000								
Province	AB	BC	MB	NB	NS	ON	QC	SK
AB	19	17	12	2	2	19	19	13
BC	17	17	12	2	2	17	17	13
MB	12	12	12	2	2	12	12	11
NB	2	2	2	2	1	2	2	2
NS	2	2	2	1	2	2	2	2
ON	19	17	12	2	2	21	21	13
QC	19	17	12	2	2	21	22	13
SK	13	13	11	2	2	13	13	13

2. ***Month hierarchy minimum support 5000 cases***

   - *Eight out of the thirteen provinces in Canada recorded over 5000 cases for at least one month each.*
   - *Five provinces (Nunavut, Prince Edward Island, Newfoundland and Labrador, Northwest Territories and Yukon) have no record of over 5000 cases per month.*

- *Ontario has the highest record of over 5000 cases per month for 19 months out of the 24 months pandemic period in the dataset.*
- *New Brunswick and Nova Scotia has 1 month each and Saskatchewan has 7 months, however, the 3 provinces have no co-occurrence pattern for any month having over 5000 cases.*

Table 6.15: Canadian COVID-19 monthly hierarchy temporal co-occurrence pattern; minsup > 5k cases

MONTH Co-occurrence Pattern where minsup > 5000								
Province	AB	BC	MB	NB	NS	ON	QC	SK
AB	13	13	4	1	1	13	13	7
BC	13	13	4	1	1	13	13	7
MB	4	4	5	1	1	5	4	3
NB	1	1	1	1	1	1	1	0
NS	1	1	1	1	1	1	1	0
ON	13	13	5	1	1	19	16	7
QC	13	13	4	1	1	16	16	7
SK	7	7	3	0	0	7	7	7

3. ***Month hierarchy minimum support 7000 cases***

- *Seven out of the thirteen provinces in Canada recorded over 7000 cases for at least one month each.*
- *six provinces (New Brunswick, Nunavut, Prince Edward Island, Newfoundland and Labrador, Northwest Territories and Yukon) have no record of over 7000 cases per month.*
- *Manitoba has 4 months, Nova Scotia has 1 month each and Saskatchewan has 4 months record of over 7,000 cases. However, the 3 provinces have no co-occurrence pattern for any month of over 7000 cases.*

Table 6.16: Canadian COVID-19 monthly hierarchy temporal co-occurrence pattern; minsup > 7k cases

MONTH Co-occurrence Pattern where minsup > 7000							
Province	AB	BC	MB	NS	ON	QC	SK
AB	13	12	4	1	13	13	4
BC	12	12	4	1	12	12	4
MB	4	4	4	1	4	4	0
NS	1	1	1	1	1	1	0
ON	13	12	4	1	18	16	4
QC	13	12	4	1	16	16	4
SK	4	4	0	0	4	4	4

4. ***Month hierarchy minimum support 10000 cases***

- *Six out of the thirteen provinces in Canada recorded over 10000 cases for at least two months each.*
- *seven provinces (New Brunswick, Nova Scotia, Nunavut, Prince Edward Island, Newfoundland and Labrador, Northwest Territories and Yukon) have no record of over 10,000 cases per month.*
- *Manitoba and Saskatchewan has 4 months record each. However, the 2 provinces have no co-occurrence pattern for any month of over 10000 cases.*

Table 6.17: Canadian COVID-19 monthly hierarchy temporal co-occurrence pattern; minsup > 10k cases

MONTH Co-occurrence Pattern where minsup > 10k						
Province	AB	BC	MB	ON	QC	SK
AB	12	11	3	12	12	2
BC	11	12	3	12	12	2
MB	3	3	3	3	3	0
ON	12	12	3	16	15	2
QC	12	12	3	15	16	2
SK	2	2	0	2	2	2

6.3.2 Evaluation on Visual Analytics of the HSTPPM Data

Experiment 6.2. *Visualization of the aforementioned data and/or patterns from the HSTM and/or HSTPPM.*

Observation 6.4. *Figures 6.5, 6.6, 6.7, 6.8, and 6.9 show the visual aspects of co-occurrence patterns mined in Observation 6.1.*

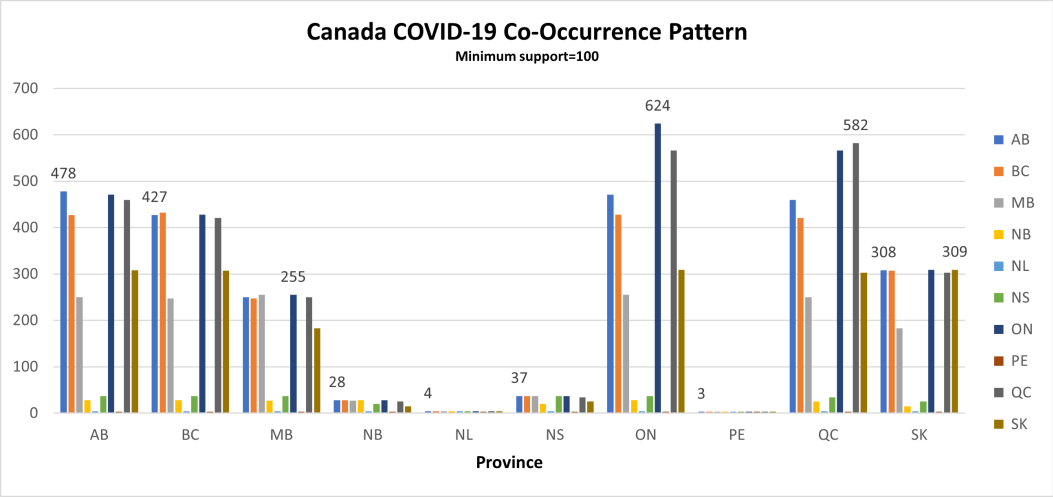


Figure 6.5: Canadian COVID-19 temporal co-occurrence pattern; minsup=100

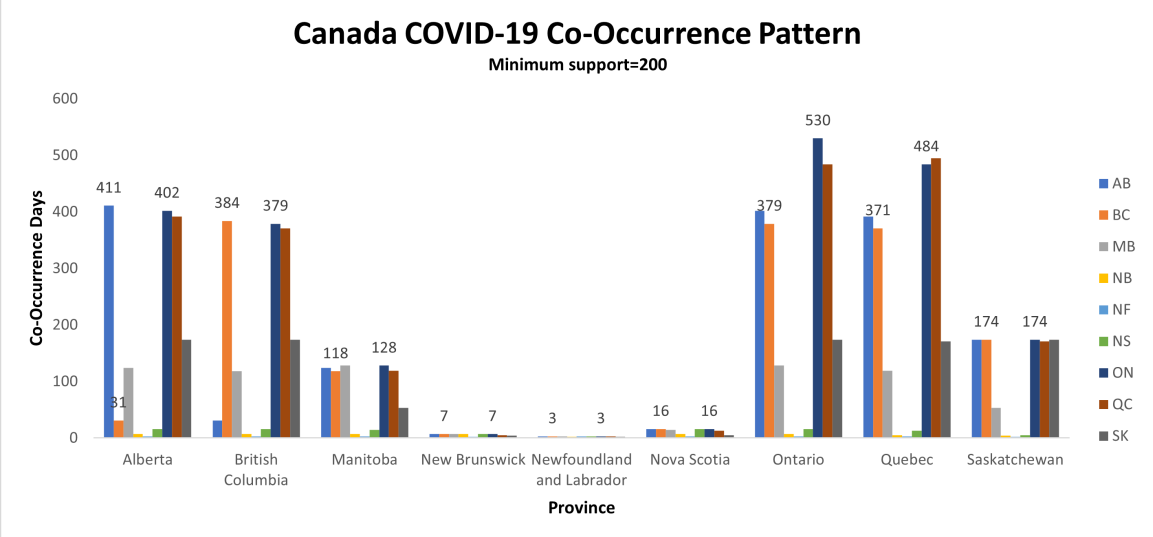


Figure 6.6: Canadian COVID-19 temporal co-occurrence pattern; minsup=200

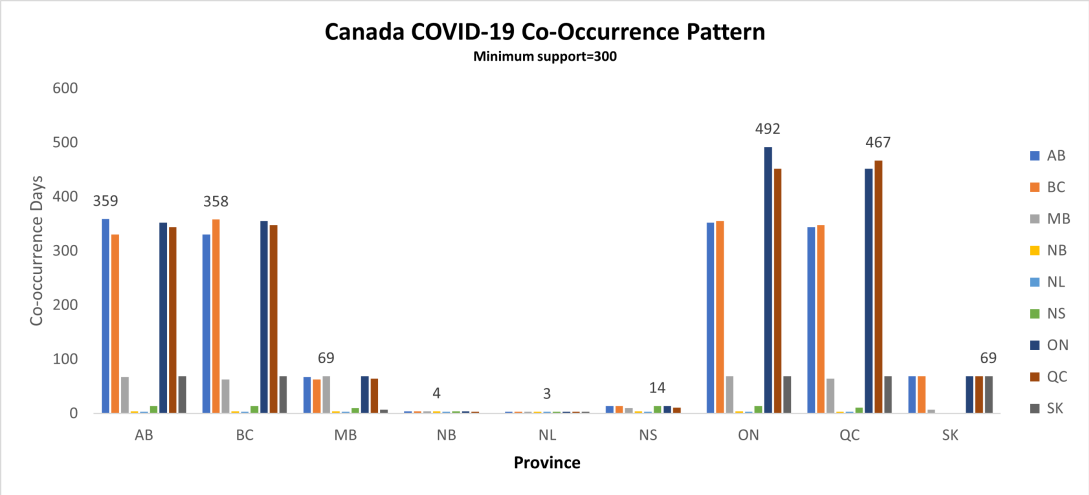


Figure 6.7: Canadian COVID-19 temporal co-occurrence pattern; minsup=300

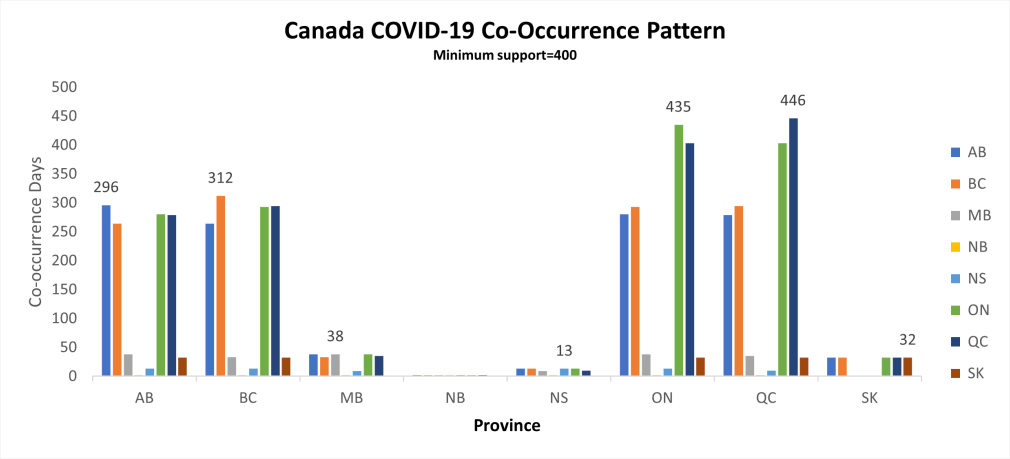


Figure 6.8: Canadian COVID-19 temporal co-occurrence pattern; minsup=400

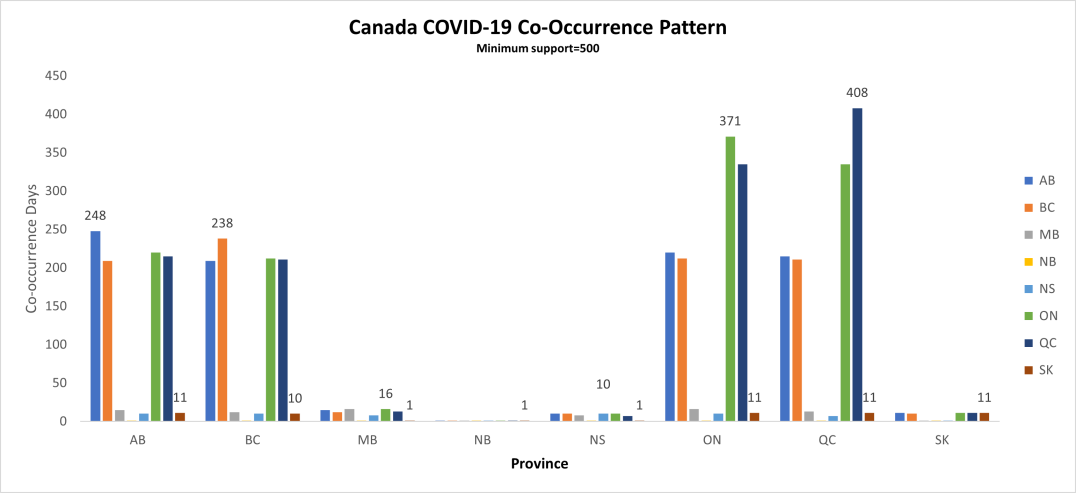


Figure 6.9: Canadian COVID-19 temporal co-occurrence pattern; minsup=500

Observation 6.5. *For City of Buffalo Summon, in terms of* ***temporal visualization****, The time series plot for the raw* **Buffalo** *dataset was not very readable without aggregation. Thus, the data were passed to the temporal hierarchy model to provide meaningful interpretation as presented in Figure 6.10 representing trends visualization.*

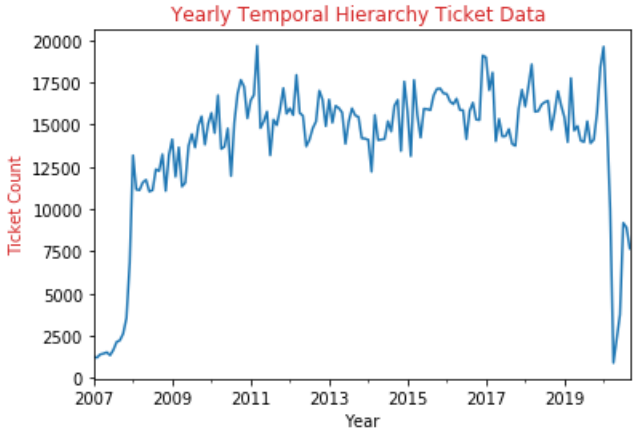


Figure 6.10: Visualize yearly trends (in a line graph) in Buffalo data

Yearly *ticket issued by the city of Buffalo from 2007 to 2020 in Figure 6.11. Observed from this column chart, the number of summon issued increased by 14% in 2009, 12% in 2010, 4% in 2011; but, decreased 2% in 2012, 2% in 2013, and 3% in 2014. It bounded back to increase by 8% in 2015, 1% in 2016. Then, it fluctuated with a decrease of 3% in 2017, an increase of 5% in 2018, and a decrease of 7% in 2019. The processing time for the hierarchy model was 0.4686 seconds and 5.7671 seconds for the raw dataset.*

Monthly *temporal hierarchy level in Figure 6.12, which shows that more summon are issued in the month of March than any other months of the year, except in 2020 when there was a general decrease in the number of summon issued through out the year from the month of March, possibly due to public restriction caused by the COVID-19 pandemic.*

Daily *temporal hierarchy level using days of week. Figure 6.13 revealed that people are more likely to get a summon in the city of Buffalo on Monday or Thursday*

Hourly *temporal visualization in Figure 6.14 shows that most summon are issued*

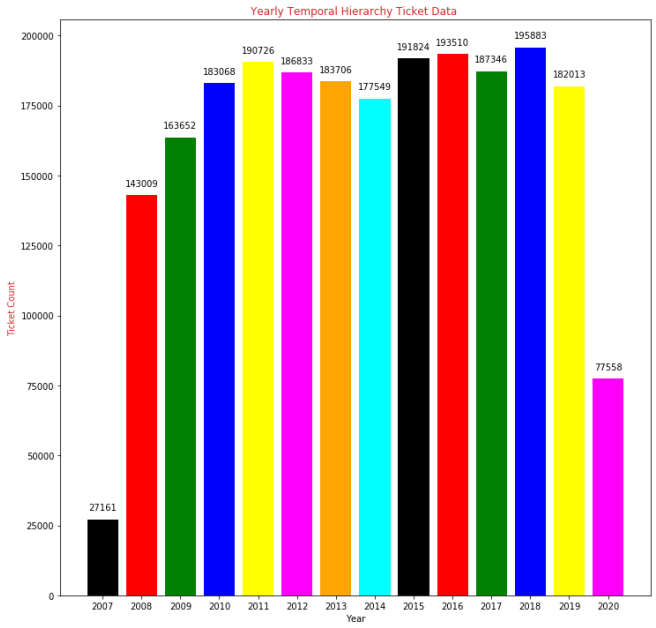


Figure 6.11: Visualize yearly patterns (in a column chart) in Buffalo

between 9 am and 12 pm having the highest recorded in 2018 between 9 am and 10 am. A second jump in number of summon is noticeable between 6 pm and 8 pm.

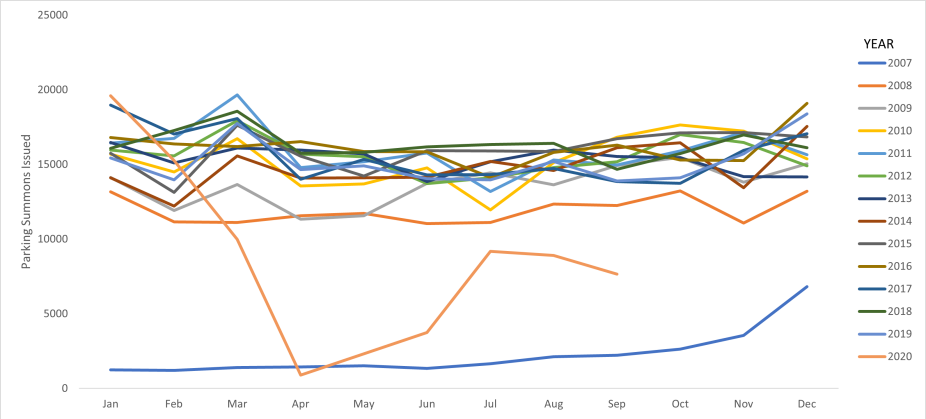


Figure 6.12: Visualize monthly trends in Buffalo data

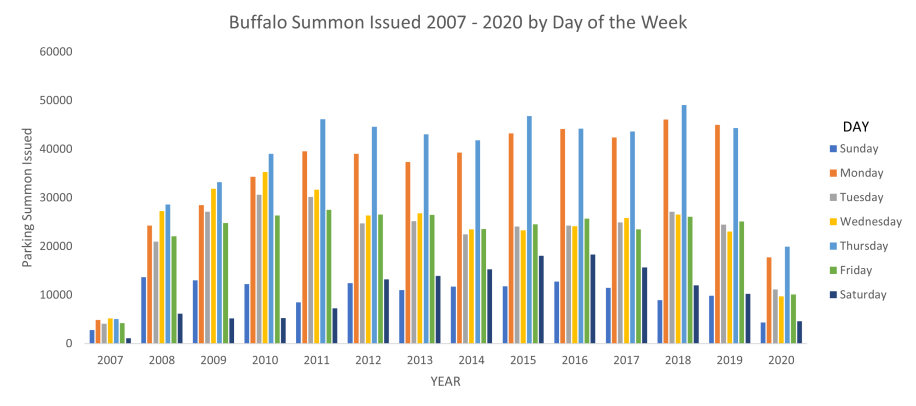


Figure 6.13: Visualize daily (days of the week) patterns in Buffalo data

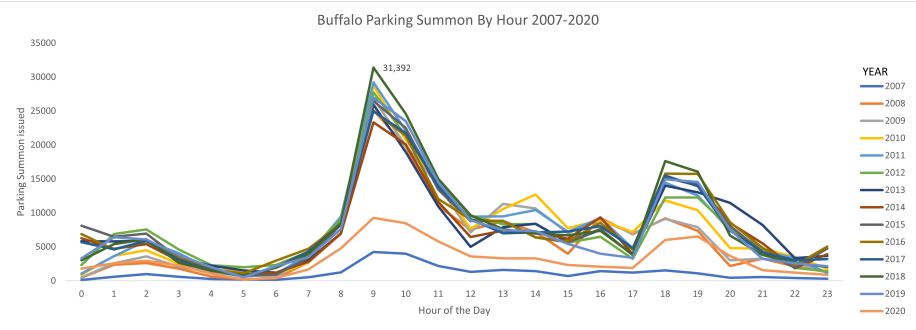


Figure 6.14: Visualize hourly trends in Buffalo data

Experiment 6.3. *For **spatial visualization**, the year 2019 Buffalo summon data were selected for the purpose of the spatial visualization using Folium library heatmap. The raw data were clumsy and showing summon issued in all streets within Buffalo*

Figure 6.15: Visualize random temporal representative points (in heat-map) in Buffalo data

Figure 6.16: Visualize centroid in Buffalo data

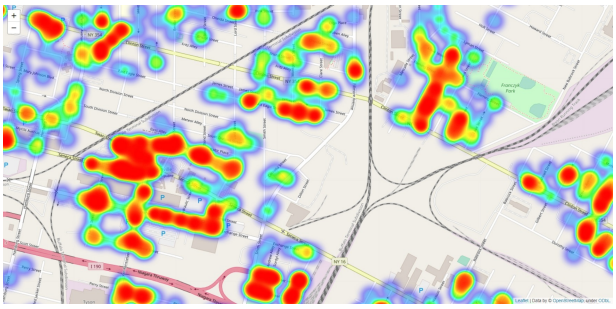


Figure 6.17: Minimum-distance points heat-map

without much difference in terms of volume.

Observation 6.6. *Figure 6.16 on centroid shows the distinct street where summon is issued for the year, whereas Figure 6.15 shows random temporal representative points vary at run time.*

Experiment 6.4. *The city of Winnipeg parking violation by-law enforcement data patterns visualization using the HSTM and/or HSTPPM for year, month, day of the*

week and hour temporal hierarchies. Note that ***Winnipeg parking ticket*** *captures parking tickets for the City of Winnipeg, MB, Canada. The dataset is 124 MB in size containing 1,617,647 parking tickets issued over a period of 10 years from January 2010 to December 2020 covering 826,929 spatial data that represent 51% of the dataset.*

Observation 6.7. *Similarly, the solution also provides privacy-preserving visualization of the* ***Winnipeg*** *parking dataset:*

- *Figure 6.18 is the column chart showing* yearly *ticket issued by the city of Winnipeg for 10 years period from January 2010 to December 2020. Similar pattern was observed as the Buffalo dataset for 2020: the number of tickets issued in 2020 decreased significantly.*

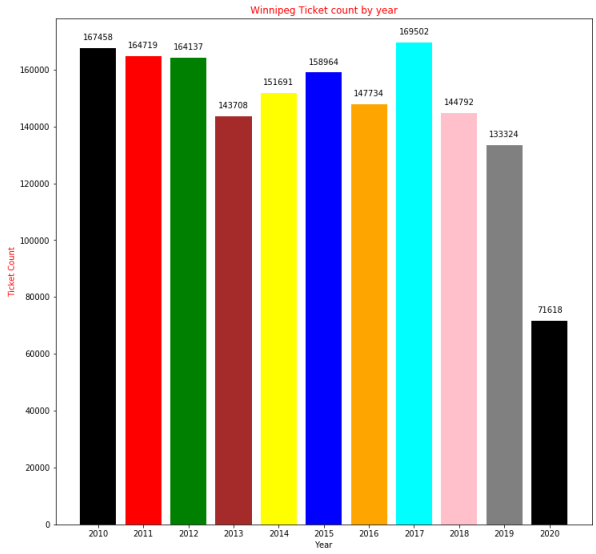


Figure 6.18: Visualize yearly patterns in Winnipeg data

- *Figure 6.19 shows the* monthly *temporal hierarchy level. Unlike Buffalo where more summon are issued consistently in the month of March, more tickets are issued in the month of December, January and February in Winnipeg indicating difference in parking patterns between the two cities.*

- *The day of week is visualized in Figure 6.20. The only consistent pattern is Monday where less tickets are issued compared to the rest of the weekdays.*

- *Winnipeg has similar* hourly *patterns as the city of Buffalo (e.g., more tickets are issued between 09:00 and 12:00 as shown in Figure 6.21).*

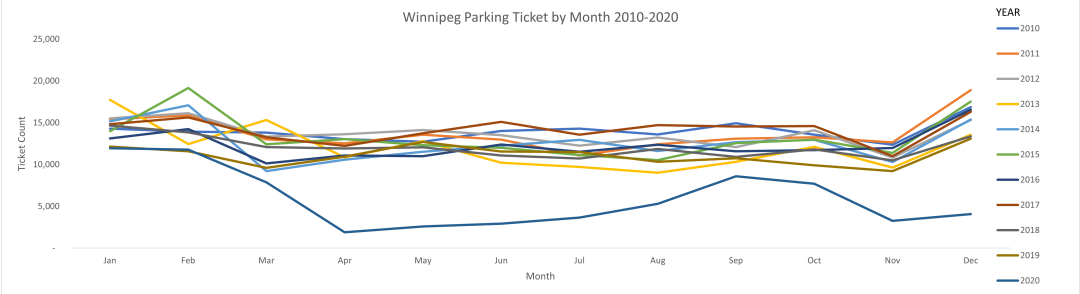


Figure 6.19: Visualize monthly trends in Winnipeg data

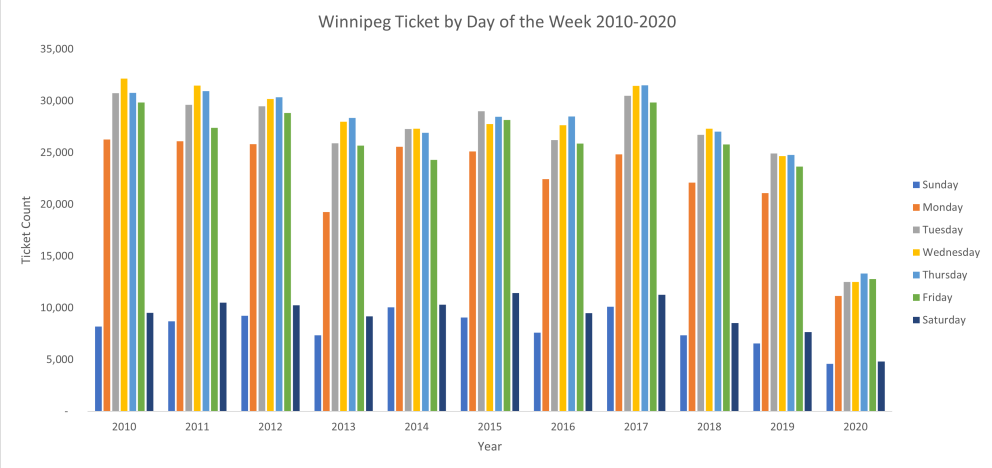


Figure 6.20: Visualize daily (days of the week) patterns in Winnipeg data

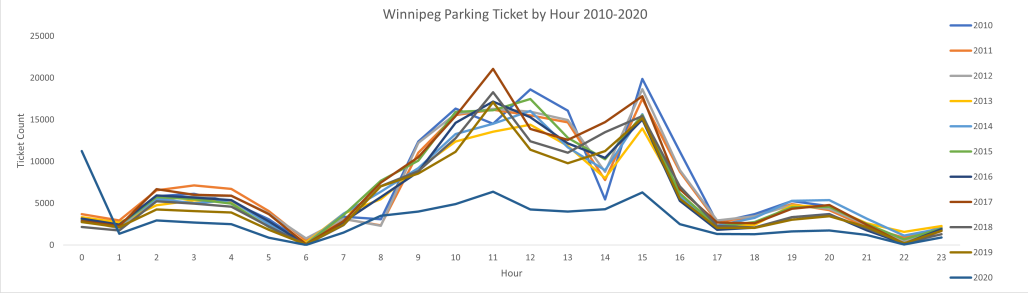


Figure 6.21: Visualize hourly trends in Winnipeg data

6.4 Summary

In this chapter, the application of the HSTPPM algorithm to two big data management tasks namely—, co-occurrence pattern mining and pattern visualization was demonstrated.

1. Privacy preserving co-occurrence pattern mining. The hierarchical spatio-temporal privacy preserving algorithm was enhanced to design co-occurrence pattern mining algorithm by adding the co-occurrence matrix step. The evaluation of the new algorithm was done using Canada COVID-19 dataset consisting of COVID-19 cases reported by Canada 10 provinces and 3 territories between January 2020 and December 2021. The day, week and month hierarchical level processing time and the co-occurrence pattern were discussed.

2. Common Pattern Visualizer: The application of the privacy preserving model on visualization of big spatio-temporal parking violation by-law enforcement was demonstrated using the city of Winnipeg and City of Buffalo parking data.

Chapter 7

Conclusions and Future Works

7.1 Conclusions

The term "Big data" has emerged over time due to the unimaginable growth in the volume of data collection resulting from the advent of the word wide web (www), internet of things (IoT) and the advancement in smaller smart devices. The challenges of managing the big data thus remains an active research area.

Hence, this thesis studied the general characteristic of big data summarizing the concept using the seven Vs namely variety, volume, velocity, veracity, visualization, validity and value. To better understand the challenges of big data, four issues were identified including (a) Complex varieties (b) Uncertain veracity inherent in big data collection (c) Questionable validity, fairness and interpretability of machine learning models used in processing big data and (d) Privacy and accessibility, a double-edged sword research challenge of achieving balance between privacy and transparency concurrently with privacy and utility.

Subsequently, the thesis set out four goals. The first goal which relates to the complex varieties issues of big data is the design of a conceptual model aiming at capturing and storing variety of big data including structured, semi-structured and unstructured data along with design of a framework for managing big data metadata beyond the data life cycle. The second goal is the design of hierarchical spatio-temporal model to support heterogeneous data linkage from the open data lake. The

third goal is privacy preserving model aiming at addressing the big data privacy issue of achieving balance between privacy and data utility. The last goal is the application of the privacy preserving models to big data management tasks including data mining and data visualization.

The thesis has provided answers to the thesis research questions discussed in Section 1.4 as follows:

Q1. How to integrate open data initiatives with data lakes?

A1. Chapter 3 presented a conceptual model to integrate the open data initiatives FAIR principle and data lake applicable to collection and storage of varieties of big data types including structured, semi-structured and unstructured data. While the open data initiative support data accessibility, data lake supports storage of structured, semi-structured and unstructured data as-is. The open data lake integrate the two concepts through architectural design comprising of three layers supporting data accessibility and reusability. The design includes a metadata framework for managing big data beyond its life cycle. The application of the conceptual model and the metadata framework was demonstrated using Amazon Web services cloud infrastructure. The dataset includes big, siloed data collected over the years by researchers through arctic expeditions, mobile science labs data, mobile art studio data and renewable-energy powered plant food production data. The metadata layer of the conceptual model provides extra benefits. First, it supports machine learning to discover relationship (a) between data and project as well as (b) among projects. Second, it aligns with the FAIR (Findable, Accessible, Interoperability, and Reusable) principles of open data concept. The conceptual model was compared with existing models such as data lake, data fabrics and data mesh and observed that the model has a distinct advantage of providing metadata collection framework which is lacking in the existing models.

Q2. How to manage open spatio-temporal data at different granularity levels? How to link heterogeneous data from different rich data sources and at different

granularity levels?

A2. The design of a hierarchical spatio-temporal model (HSTM) comprising of (a) temporal hierarchy with extension to time generalization for low level granularity), (b) location hierarchy and (c) temporal representative points was presented in Chapter 4. The HSTM supports big data analytics on temporal granularity harnessing the different level of temporal hierarchies; supports various level of granularity analytics on different location hierarchies and the temporal representative points enable the HSTM to represent a collection of points within a specific level of temporal and/or location hierarchy. The best approach to represent group of spatial points at any hierarchical level was investigated and observed the demerit of a centroid approach, though the centroid provides high privacy protection, it is not a real point. Thus, an enhanced random approach called "medroid" was introduced. The medroid is a random point closer to the centroid and it is guaranteed to be a real point within the group of observations and also provide privacy protection. The application and effectiveness of the proposed approach was demonstrated by linking four different dataset from the city of Winnipeg open data portal including fire paramedic emergency call logs, substance-use data, Naloxone administration for the treatment of opioid overdose patient and parking violation by-law enforcement data. The model is applicable to linking of structured, semi-structured and unstructured data stored in the open data lake by harnessing the temporal and location metadata collected from heterogeneous data sources.

Q3. How to preserve privacy of spatio-temporal data? How to do so for privacy-preserving publishing? How to do so far COVID-19 contact tracing?

A3. The thesis focused on approach to preserving privacy of big data using the concept of time and space correlation. In Chapter 5, a hierarchical privacy preserving model aiming at protecting the temporal and location data in the big dataset was designed. The proposed approach is a bottom-up aggregation extended to the hierarchy of time and space such that the disclosure risk is

minimized to the trends and pattern in the aggregated dataset without revealing the individual information. The application of the privacy preserving models to data publishing and contact tracing was demonstrated using bylaw enforcement parking violation dataset and South Korea COVID-19 contact tracing dataset.

Q4. How to mine interesting patterns—such as co-occurrence patterns—from open spatio-temporal data at different granularity levels? How to visualize them?

A4. In Chapter 6, the hierarchical privacy preserving spatio-temporal model (HSTPPM) was enhanced to design co-occurrence pattern mining algorithm. The algorithm was applied to spatio-temporal co-occurrence pattern mining of Canada COVID-19 dataset and demonstrated the effectiveness of the algorithm on different temporal hierarchy of day, week and month. In addition, The application and effectiveness of the HSTPPM on big data visualization was demonstrated using parking violation by-law enforcement dataset.

In conclusion, in this thesis, I have designed big data management models aiming at addressing big data issues of complex varieties, privacy and accessibility of data. The application and effectiveness of the models have been demonstrated using varieties of real life dataset including COVID-19 dataset, bylaw enforcement parking dataset as well as paramedic emergency, substance-use and Naloxone administration for the treatment of Opioid overdose dataset. The complex variety open data lake conceptual model is applicable to any big data and the metadata framework supports machine learning discovery of relationship among variety of big data types. The privacy preserving model is applicable to big data co-occurrence pattern mining, big data publishing, big data integration, big data compression and big data visualization.

7.2 Future Work

In this section, some future considerations to extend this thesis work are discussed.

7.2.1 Explainable Metadata Machine Learning

The challenge of capturing complex variety of big data type from heterogeneous sources and reusability has been addressed in the current research by designing the open data lake architecture and metadata framework to support machine learning of interrelationship among varieties of big data and among projects. Thus, extending this thesis to develop explainable metadata machine learning algorithm to further enhance reliability and validity of machine learning models for big data processing task including data mining and data publishing is of future consideration.

7.2.2 Privacy Preserving Knowledge Graph

Knowledge graph has been defined in this thesis as an integrated framework for large scale knowledge acquisition from heterogeneous sources, knowledge representation and reuse based on graph architecture [OLC23]. The second future research direction is the extension of the privacy preserving model to preserving graph embedding data and the application to big data management tasks such as data mining, data publishing, data integration and data visualization. Graph neural network has recently evolved to process complex interactions in the social network, biological data and telecommunication data embedded in graph structure. However, the major challenge of preserving privacy is an active research area. Hence, extending the current architecture and the metadata framework to knowledge graph using graph data structure is one research area for future consideration. Another area of interest is the extension of this thesis work of bottom-up aggregation approach to preserving privacy of data on graph temporal hierarchy.

7.2.3 Big Data Co-occurrence Pattern Mining on Graph Structure

Frequent itemset mining consideration is often on the entire dataset without consideration to variety of dataset from heterogeneous sources and correlation of temporal and spatial attributes. Hence, an area of interest is to study how to leverage the theoretical connection of statistics, machine learning and data mining fields to develop a graph structure model for mining co-occurrence pattern in heterogeneous datasets.

7.2.4 Heterogeneous Complex Queries Integration and Optimization

One of the challenges identified for big data lake in Section 2.2.11.2 is the issue of queries rewriting paradigm. A common approach to this issue is the creation of multiple queries from the data lakes to answer users' questions for business decision making. However, these independent queries may be related and integrating such queries may result in other interesting or rare patterns that are of important to decision making. Hence, an area of interest is to investigate the theoretical techniques for integrating multiple complex queries as well as techniques for optimizing the performance of the integrated query and its application to big data analytics and visualization.

Bibliography

[ABCP13] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, pages 901–914, 2013.

[ABK+07] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *Proceedings of the International Semantic Web Conference 2007*, pages 722–735. Springer, 2007.

[AC14] Gergely Acs and Claude Castelluccia. A case study: Privacy preserving release of spatio-temporal density in paris. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1679–1688, 2014.

[AG08] Renzo Angles and Claudio Gutierrez. Survey of graph database models. *ACM Computing Surveys (CSUR)*, 40(1):1–39, 2008.

[ALDC22] Micah M Alvord, Fengyu Lu, Boyang Du, and Chia-An Chen. Big data fabric architecture: How big data and data management frameworks converge to bring a new generation of competitive advantage for enterprises. *Enterprise Architecture Professional Journal*, 2022.

[ALMK16] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias risk assessments in criminal sentencing. *ProPublica*, May 23, 2016.

[Bac69] Charles W Bachman. Data structure diagrams. *ACM SIGMIS Database: The DATABASE for Advances in Information Systems*, 1(2):4–10, 1969.

[Bak87] René Ronald Bakker. *Knowledge graphs: representation and structuring of scientific knowledge*. PhD thesis, University of Twente, Netherlands, 1987.

[BBA17] Mahfoud Bala, Omar Boussaid, and Zaia Alimazighi. A fine-grained distribution approach for etl processes in big data environments. *Data & Knowledge Engineering*, 111:114–136, 2017.

[BC12] Danah Boyd and Kate Crawford. Critical questions for big data: Provocations for a cultural, technological, and scholarly phenomenon. *Information, communication & society*, 15(5):662–679, 2012.

[BCMT13] Roi Blanco, Berkant Barla Cambazoglu, Peter Mika, and Nicolas Torzec. Entity recommendations in web search. In *Proceedings of the 12th International Semantic Web Conference 2013*, pages 33–48. Springer, 2013.

[BDH96] C Bradford Barber, David P Dobkin, and Hannu Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483, 1996.

[BFW00] Peter Buneman, Wenfei Fan, and Scott Weinstein. Path constraints in semistructured databases. *Journal of Computer and System Sciences*, 61(2):146–193, 2000.

[BKKH23] Jan Bode, Niklas Kühl, Dominik Kreuzberger, and Sebastian Hirschl. Data mesh: Motivational factors, challenges, and best practices. *arXiv preprint arXiv:2302.01713*, 2023.

[Bol13] Béla Bollobás. *Modern graph theory*. Springer Science & Business Media, 2013.

[BOL19] John Brackenbury, PY O'Shaughnessy, and Yan-Xia Lin. Protecting the privacy of smart meter data: the differential privacy approach and the

multiplicative noise approach. Technical report, University of Wollongong, Australia, 2019.

[BPS+97] Tim Bray, Jean Paoli, C Michael Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible markup language (XML). *World Wide Web Journal*, 2(4):27–66, 1997.

[Bun97] Peter Buneman. Semistructured data. In *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 117–121, 1997.

[CASL17] Ashis Kumar Chanda, Chowdhury Farhan Ahmed, Md Samiullah, and Carson K. Leung. A new framework for mining weighted periodic patterns in time series databases. *Expert Systems with Applications*, 79:207–224, 2017.

[CBK+10] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam Hruschka, and Tom Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of the AAAI Conference on Artificial Intelligence 2010*, pages 1306–1313, 2010.

[CERE90] Bogdan Czejdo, Ramez Elmasri, Marek Rusinkiewicz, and David W. Embley. A graphical data manipulation language for an extended entity-relationship model. *Computer*, 23(3):26–36, 1990.

[Che76] Peter Pin-Shan Chen. The entity-relationship model—toward a unified view of data. *ACM transactions on database systems (TODS)*, 1(1):9–36, 1976.

[Che02] Peter Chen. Entity-relationship modeling: historical events, future trends, and lessons learned. *Software pioneers: contributions to software engineering*, pages 296–310, 2002.

[CI19] Silvia Chiappa and William S Isaac. A causal bayesian networks viewpoint on fairness. *Privacy and Identity Management. Fairness, Account-*

ability, and Transparency in the Age of Big Data: 13th IFIP WG 9.2, 9.6/11.7, 11.6/SIG 9.2.2 International Summer School 2018, Revised Selected Papers, pages 3–20, 2019.

[CLSO22] Alfredo Cuzzocrea, Carson K. Leung, Selim Soufargi, and Anifat M Olawoyin. The emerging challenges of big data lakes, and a real-life framework for representing, managing and supporting machine learning on big arctic data. In *Proceedings of the International Conference on Intelligent Networking and Collaborative Systems*, pages 161–174. Springer, 2022.

[CLZW16] Xi Chen, Jianwei Li, Zeyu Zheng, and Tianran Wang. Degree of integration into social networks (disn) evaluation model based on micro-blogging platforms and information dissemination prediction. In *Proceedings of the International Conference on Computer and Information Technology (CIT)*, pages 172–176. IEEE, 2016.

[Cod70] Edgar F Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[Con15] UniProt Consortium. Uniprot: a hub for protein information. *Nucleic acids research*, 43(D1):D204–D212, 2015.

[CSN+14] Mandy Chessell, Ferd Scheepers, Nhan Nguyen, Ruud van Kessel, and Ron van der Starre. Governing and managing big data for analytics and decision makers. *IBM Redguides for Business Leaders*, 2014.

[CWD+18] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.

[CWZ+20] Zhe Chen, Yuehan Wang, Bin Zhao, Jing Cheng, Xin Zhao, and Zongtao Duan. Knowledge graph completion: A review. *IEEE Access*, 8:192435–192456, 2020.

[CYXX17] Yang Cao, Masatoshi Yoshikawa, Yonghui Xiao, and Li Xiong. Quantifying differential privacy under temporal correlations. In *Proceedings of the 33rd International Conference on Data Engineering (ICDE)*, pages 821–832. IEEE, 2017.

[Deh22] Zhamak Dehghani. *Data Mesh.* O'Reilly Media Inc., 2022.

[DGH+14] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 601–610, 2014.

[DGLT23] Jasmin K. Dhaliwal, Megan E. Galbraith, Carson K. Leung, and Da Tan. A data discovery and visualization tool for visual analytics of time series in digital agriculture. In *Proceedings of the 27th International Conference Information Visualisation (IV)*, pages 268–271. IEEE, 2023.

[DKS+24] Butros M Dahu, Solaiman Khan, Ammar Ali Salman, Youssef Michal Andraws, A Abo Daken, and Ahmad Aburayya. Epidemiological analysis of vaccination strategies and demographic patterns in COVID-19 cases in the midwest region of the United States. *National Journal of Community Mediine*, 14(1):62–71, 2024.

[DLP18] Chris Donahue, Bo Li, and Rohit Prabhavalkar. Exploring speech enhancement with generative adversarial networks for robust speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5024–5028. IEEE, 2018.

[DSD+11] N. Dovrolis, T. Stefanut, S. Dietze, H. Q. Yu, C. Valentine, and Kaldoudi E. Semantic annotation and linking of medical educational resources. In *Proceedings of the FIfth European Conference of the International Federation for Medical and Biological Engineering. Heidelberg*, pages 1400–1403, Berlin, 2011. Springer.

[EW16] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48(1-4):2, 2016.

[Fan15] Huang Fang. Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem. In *Proceedings of the International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pages 820–824. IEEE, 2015.

[FBMR18] Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web*, 9(1):77–129, 2018.

[FKV+18] Antonio Fabregat, Florian Korninger, Guilherme Viteri, Konstantinos Sidiropoulos, Pablo Marin-Garcia, Peipei Ping, Guanming Wu, Lincoln Stein, Peter D'Eustachio, and Henning Hermjakob. Reactome graph database: Efficient access to complex pathway data. *PLOS Computational Biology*, 14(1):e1005968, 2018.

[GGH+19] Corinna Giebler, Christoph Gröger, Eva Hoos, Holger Schwarz, and Bernhard Mitschang. Leveraging the data lake: Current state and challenges. In *Proceedings of the International Conference on Big Data Analytics and Knowledge Discovery 2019*, pages 179–188. Springer, 2019.

[GLH+18] Yuqing Gao, Jisheng Liang, Benjamin Han, Mohamed Yakout, and Ahmed Mohamed. Building a large-scale, accurate and fresh knowledge graph. In *ACM KDD 2018 Tutorial*, pages 1939–1374, 2018.

[GPM+20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[GSW+17] Zhitao Guan, Guanlin Si, Jun Wu, Liehuang Zhu, Zijian Zhang, and Yinglong Ma. Utility-privacy tradeoff based on random data obfuscation in internet of energy. *IEEE Access*, 5:3250–3262, 2017.

[HGQ16] Rihan Hai, Sandra Geisler, and Christoph Quix. Constance: An intelligent data lake system. In *Proceedings of the International Conference on Management of Data (SIGMOD) 2016*, pages 2097–2100, 2016.

[HMS+17] Dominik Hornung, Claudia Müller, Irina Shklovski, Timo Jakobi, and Volker Wulf. Navigating relationships and boundaries: Concerns around ict-uptake for elderly people. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 7057–7069, 2017.

[HPT22] Jiawei Han, Jian Pei, and Hanghang Tong. *Data mining: concepts and techniques, 4th edition*. Morgan Kaufmann, 2022.

[HWH+24] Christine Mary Hallinan, Roger Ward, Graeme K Hart, Clair Sullivan, Nicole Pratt, Ashley P Ng, Daniel Capurro, Anton Van Der Vegt, Siaw-Teng Liaw, Oliver Daly, et al. Seamless EMR data access: Integrated governance, digital health and the OMOP-CDM. *BMJ Health & Care Informatics*, 31(1), 2024.

[HWL12] Li-Yung Ho, Jan-Jan Wu, and Pangfeng Liu. Distributed graph database for large-scale social computing. In *Proceedings of the fifth International Conference on Cloud Computing*, pages 455–462. IEEE, 2012.

[HWW15] Ron Henkel, Olaf Wolkenhauer, and Dagmar Waltemath. Combining computational models, semantic annotations and simulation experiments in a graph database. *Database*, 2015:bau130, 2015.

[HZK13] Xingze He, Xinwen Zhang, and C-C Jay Kuo. A distortion-based approach to privacy-preserving metering in smart grids. *IEEE Access*, 1:67–78, 2013.

[Inm05] William H Inmon. *Building the data warehouse*. John wiley & sons, 2005.

[Inm16] Bill Inmon. *Data Lake Architecture: Designing the Data Lake and avoiding the garbage dump*. Technics publications, 2016.

[JdJ+20] Annika Jacobsen, Ricardo de Miranda Azevedo, Nick Juty, Dominique Batista, Simon Coles, Ronald Cornet, Mélanie Courtot, Mercè Crosas, Michel Dumontier, Chris T Evelo, et al. FAIR principles: interpretations and implementation considerations. *Data intelligence*, 2(1-2):10–29, 2020.

[JL15] Fan Jiang and Carson K. Leung. A data analytic algorithm for managing, querying, and processing uncertain big data in cloud environments. *Algorithms*, 8(4):1175–1194, 2015.

[JLP16] Fan Jiang, Carson Leung, and Adam G. M. Pazdor. Web page recommendation based on bitwise frequent pattern mining. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 632–635, 2016.

[JLT12] Fan Jiang, Carson Kai-Sang Leung, and Syed K. Tanbeer. Finding popular friends in social networks. In *Proceedings of the Second International Conference on Cloud and Green Computing*, pages 501–508. IEEE, 2012.

[JM17] Tomcy John and Pankaj Misra. *Data lake for enterprises*. Packt Publishing Ltd, 2017.

[JSB+13] Kaifeng Jiang, Dongxu Shao, Stéphane Bressan, Thomas Kister, and Kian-Lee Tan. Publishing trajectories with differential privacy guarantees. In *Proceedings of the 25th International Conference on Scientific and Statistical Database Management*, pages 1–12, 2013.

[JWS+20] Nick Juty, Sarala M Wimalaratne, Stian Soiland-Reyes, John Kunze, Carole A Goble, and Tim Clark. Unique, persistent, resolvable: Identifiers as the foundation of FAIR. *Data Intelligence*, 2(1-2):30–39, 2020.

[Ken23] Ben Kenwright. Convex hulls: Surface mapping onto a sphere. *arXiv preprint arXiv:2304.04079*, 2023.

[KG00] Minoru Kanehisa and Susumu Goto. Kegg: kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28(1):27–30, 2000.

[KHM+16] Sebastian Krause, Leonhard Hennig, Andrea Moro, Dirk Weissenborn, Feiyu Xu, Hans Uszkoreit, and Roberto Navigli. Sar-graphs: A language resource connecting linguistic knowledge with semantic relations from knowledge graphs. *Journal of Web Semantics*, 37:112–131, 2016.

[Kim90] Won Kim. Object-oriented databases: Definition and research directions. *IEEE Transactions on Knowledge and Data Engineering*, 2(3):327–341, 1990.

[KM00] Ralph Kimball and Richard Merz. *The data webhouse toolkit: Building the Web-enabled data warehouse*. John Wiley& Sons, 2000.

[Knu14] Donald E Knuth. *Art of computer programming, volume 2: Seminumerical algorithms*. Addison-Wesley Professional, 2014.

[Kre89] David M Kreps. Nash equilibrium. In *Game Theory*, pages 167–177. Springer, 1989.

[KRP+17] Niki Kilbertus, Mateo RojasCarulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding discrimination through causal reasoning. In *Proceedings of 31st Conference on Neural Information Processing Systems (NIPS 2017)*, pages 656–666, 2017.

[KS14] Shiva P Kasiviswanathan and Adam Smith. On the 'semantics' of differential privacy: A bayesian formulation. *Journal of Privacy and Confidentiality*, 6(1), 2014.

[KSK20] Stefan M Kostić, Mirjana I Simić, and Miroljub V Kostić. Social network analysis and churn prediction in telecommunications using graph theory. *Entropy*, 22(7):753, 2020.

[KVKZ21] Kirill Krinkin, AI Vodyaho, IA Kulikov, and NA Zhukova. The method of inductive synthesis of hierarchical knowledge graphs of telecommunication networks based on statistical data. *Procedia Computer Science*, 186:571–579, 2021.

[KW03] J Kim and W Winkler. Multiplicative noise for masking continuous data. *Statistics*, 1(9), 2003.

[LAA+23] Abdalwali Lutfi, Mahmaod Alrawad, Adi Alsyouf, Mohammed Amin Almaiah, Ahmad Al-Khasawneh, Akif Lutfi Al-Khasawneh, Ahmad Farhan Alshira'h, Malek Hamed Alshirah, Mohamed Saad, and Nahla Ibrahim. Drivers and impact of big data analytic adoption in the retail industry: A quantitative investigation applying structural equation modeling. *Journal of Retailing and Consumer Services*, 70:103129, 2023.

[LAL+21] Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong, Clark Barrett, Mykel J Kochenderfer, et al. Algorithms for verifying deep neural networks. *Foundations and Trends® in Optimization*, 4(3-4):244–404, 2021.

[LBH+19] Carson K. Leung, Peter Braun, Calvin S. H. Hoi, Joglas Souza, and Alfredo Cuzzocrea. Urban analytics of big transportation data for supporting smart cities. In *Proceedings of the International Conference on Big Data Analytics and Knowledge Discovery*, pages 24–33. Springer, 2019.

[LEM+20] Carson K. Leung, Jonathan D. Elias, Shael M. Minuk, A. Roy R. de Jesus, and Alfredo Cuzzocrea. An innovative fuzzy logic-based machine learning algorithm for supporting predictive analytics on big transportation data. In *Proceedings of the 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1905–1912. IEEE, 2020.

[Leu23] Carson K. Leung. Big data visualization of association rules and frequent patterns. In *Encyclopedia of Data Science and Machine Learning*, pages 1284–1298. IGI Global, 2023.

[Leu24] Carson K. Leung. Biomedical informatics: State of the art, challenges, and opportunities. *BioMedInformatics*, 4(1):89–97, 2024.

[LFM+21] Carson K. Leung, Daryl L. X. Fung, Daniel Mai, Qi Wen, Jason Tran, and Joglas Souza. Explainable data analytics for disease and healthcare informatics. In *Proceedings of the 25th International Database Engineering & Applications Symposium*, pages 65–74, 2021.

[LH13] Carson Kai-Sang Leung and Yaroslav Hayduk. Mining frequent patterns from uncertain data with MapReduce for big data analytics. In *Proceedings of the International Conference on Database Systems for Advanced Applications 2013*, pages 440–455. Springer, 2013.

[Lin85] Tok-Wang Ling. Normal form for entity-relationship diagrams. Technical report, National University of Singapore, 1985.

[LJ15] Carson Kai-Sang Leung and Fan Jiang. Big data analytics of social networks for the discovery of "following" patterns. In *Proceedings of the 17th International Conference on Big Data Analytics and Knowledge Discovery (DaWaK 2015)*, pages 123–135. Springer, 2015.

[LK21] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 650–665. Springer, 2021.

[LLV07] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Proceedings of the 23rd International Conference on Data Engineering*, pages 106–115. IEEE, 2007.

[LLY+21] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, 33:6999–7019, 2021.

[LMJ14] Carson Kai-Sang Leung, Richard Kyle MacKinnon, and Fan Jiang. Distributed uncertain data mining for frequent patterns satisfying anti-monotonic constraints. In *Proceedings of the 28th International Conference on Advanced Information Networking and Applications Workshops*, pages 1–6. IEEE, 2014.

[LMKA16] Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. How we analyzed the compas recidivism algorithm. *ProPublica (5 2016)*, 9(1):3–3, 2016.

[LMT14] Carson Kai-Sang Leung, Richard Kyle MacKinnon, and Syed K Tanbeer. Fast algorithms for frequent itemset mining from uncertain data. In *Proceedings of the 2014 IEEE International Conference on Data Mining*, pages 893–898. IEEE, 2014.

[LMW14] Carson Kai-Sang Leung, Richard Kyle MacKinnon, and Yang Wang. A machine learning approach for stock price prediction. In *Proceedings of the 18th International Database Engineering & Applications Symposium*, pages 274–277, 2014.

[LÖ18] Ling Liu and M Tamer Özsu. *Encyclopedia of Database Systems, 2nd Ed.* Springer, 2018.

[LOW21] Carson K. Leung, Anifat M Olawoyin, and Qi Wen. Privacy-preserving healthcare analytics of trajectory data. In *Proceedings of the Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*, pages 414–420. Springer, 2021.

[MCS21] Inês Machado, Carlos Costa, and Maribel Yasmina Santos. Data-driven information systems: the data mesh paradigm shift. In *Proceedings of the International Conference on Information Systems Development (ISD)*, 2021.

[MCS22a] Inês Araújo Machado, Carlos Costa, and Maribel Yasmina Santos. Advancing data architectures with data mesh implementations. In *Proceedings of the Intelligent Information Systems: CAiSE Forum 2022*, pages 10–18. Springer, 2022.

[MCS22b] Inês Araújo Machado, Carlos Costa, and Maribel Yasmina Santos. Data mesh: concepts and principles of a paradigm shift in data architectures. *Procedia Computer Science*, 196:263–271, 2022.

[MFDW17] Dominik Moritz, Danyel Fisher, Bolin Ding, and Chi Wang. Trust, but verify: Optimistic visualizations of approximate queries for exploring big data. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 2904–2915, 2017.

[Mil98] George A Miller. *WordNet: An electronic lexical database*. MIT Press, 1998.

[MKG+20] Raha Moraffah, Mansooreh Karami, Ruocheng Guo, Adrienne Raglin, and Huan Liu. Causal interpretability for machine learning-problems, methods and evaluation. *ACM SIGKDD Explorations*, 22(1):18–33, 2020.

[MKGV07] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3–es, 2007.

[ML16] Cedrine Madera and Anne Laurent. The next information architecture evolution: the data lake wave. In *Proceedings of the 8th International Conference on Management of Digital EcoSystems*, pages 174–180, 2016.

[MMC+18] Mario Mezzanzanica, Fabio Mercorio, Mirko Cesarini, Vincenzo Moscato, and Antonio Picariello. Graphdblp: A system for analysing networks of computer scientists through graph databases. *Multimedia Tools and Applications*, 77(14):18657–18688, 2018.

[MMS13] Peter Macko, Daniel Margo, and Margo Seltzer. Performance introspection of graph databases. In *Proceedings of the 6th International Systems and Storage Conference*, pages 1–10, 2013.

[MS92] Victor M Markowitz and Arie Shoshani. Representing extended entity-relationship structures in relational databases: A modular approach. *ACM Transactions on Database Systems (TODS)*, 17(3):423–464, 1992.

[MST20] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency 2020*, pages 607–617, 2020.

[MT17] Antonio Maccioni and Riccardo Torlone. Crossing the finish line faster when paddling the data lake with kayak. *Proceedings of the Very Large Database (VLDB) Endowment*, 10(12):1853–1856, 2017.

[MYZ+21] Zijun Mao, Hong Yao, Qi Zou, Weiting Zhang, Ying Dong, et al. Digital contact tracing based on a graph database algorithm for emergency management during the COVID-19 epidemic: Case study. *JMIR mHealth and uHealth*, 9(1):e26836, 2021.

[Nav92] Shamkant B Navathe. Evolution of data modeling for databases. *Communications of the ACM*, 35(9):112–123, 1992.

[NEL86] Shamkant Navathe, Ramez Elmasri, and James Larson. Integrating user views in database design. *Computer*, 19(1):50–62, 1986.

[NH08] Sri Nurdiati and Cornelis Hoede. 25 years development of knowledge graph theory: the results and the challenge. *Memorandum*, 1876(2):1–10, 2008.

[NKDC15] Daniel Nicoara, Shahin Kamali, Khuzaima Daudjee, and Lei Chen. Hermes: Dynamic partitioning for distributed social network graph

databases. In *Proceedings of the 18th International Conference on Extending Database Technology (EDBT) 2015*, pages 25–36, 2015.

[NP12] Roberto Navigli and Simone Paolo Ponzetto. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250, 2012.

[NSR+19] Linh Kieu Nguyen, Chanyoung Song, Joonghyun Ryu, Phan Thanh An, Nam-Dũng Hoang, and Deok-Soo Kim. Quickhulldisk: A faster convex hull algorithm for disks. *Applied Mathematics and Computation*, 363:124626, 2019.

[OC18] Anifat Olawoyin and Yangjuin Chen. Predicting the future with artificial neural network. *Procedia Computer Science*, 140:383–392, 2018.

[OL22] Anifat M. Olawoyin and Carson K. Leung. Preserving privacy integration and mining for big temporal co-occurrence patterns. In *Proceedings of the International Conference on Big Data 2022*, pages 3959–3968. IEEE, 2022.

[OLC20a] Anifat M. Olawoyin, Carson K. Leung, and Ratna Choudhury. Privacy-preserving spatio-temporal patient data publishing. In *Proceedings of the International Conference on Database and Expert Systems Applications*, pages 407–416. Springer, 2020.

[OLC20b] Anifat M. Olawoyin, Carson K. Leung, and Alfredo Cuzzocrea. Preserving privacy of temporal big data. In *Proceedings of the International Conference on Big Data 2020*, pages 4042–4051. IEEE, 2020.

[OLC21a] Anifat M. Olawoyin, Carson K. Leung, and Alfredo Cuzzocrea. Open data lake to support machine learning on arctic big data. In *Proceedings of the International Conference on Big Data 2021*, pages 5215–5224. IEEE, 2021.

[OLC21b] Anifat M. Olawoyin, Carson K. Leung, and Alfredo Cuzzocrea. Privacy-preserving publishing and visualization of spatial-temporal information. In *Proceedings of the International Conference on Big Data 2021*, pages 5420–5429. IEEE, 2021.

[OLC23] Anifat M. Olawoyin, Carson K. Leung, and Alfredo Cuzzocrea. Evolution of big data models from hierarchical models to knowledge graphs. In *Proceedings of the 47th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 5215–5224. IEEE, 2023.

[OLHC23] Anifat M. Olawoyin, Carson K. Leung, Connor C. J. Hryhoruk, and Alfredo Cuzzocrea. Big data management for machine learning from big data. In *Proceedings of the 37th International Conference on Advanced Information Networking and Applications (AINA-2023), Volume 1*, pages 393–405. Springer, 2023.

[OLW21] Anifat M. Olawoyin, Carson K. Leung, and Qi Wen. Privacy preservation of COVID-19 contact tracing data. In *Proceedings of the 20th International Conference on Ubiquitous Computing and Communications (IUCC/CIT/DSCI/SmartCNS 2021)*, pages 288–295. IEEE, 2021.

[Pau17] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508, 2017.

[PDR21] Debachudamani Prusti, Daisy Das, and Santanu Kumar Rath. Credit card fraud detection technique by applying graph database model. *Arabian Journal for Science and Engineering*, 46(9):1–20, 2021.

[PKM22] Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Cok: A survey of privacy challenges in relation to data meshes. In *Proceedings of the 33rd International Conference,Database and Expert Systems Applications (DEXA 2022)*, pages 85–102. Springer, 2022.

[PM88] Joan Peckham and Fred Maryanski. Semantic data models. *ACM Computing Surveys (CSUR)*, 20(3):153–189, 1988.

[PVS+18] Charles Perin, Romain Vuillemot, Charles D Stolper, John T Stasko, Jo Wood, and Sheelagh Carpendale. State of the art of sports data visualization. *Computer Graphics Forum*, 37(3):663–686, 2018.

[PWS20] Thomas PellissierTanon, Gerhard Weikum, and Fabian Suchanek. Yago 4: A reason-able knowledge base. In *Proceedings of the European Semantic Web Conference 2020*, pages 583–596. Springer, 2020.

[RAL19] Md Mahmudur Rahman, Chowdhury Farhan Ahmed, and Carson Kai-Sang Leung. Mining weighted frequent sequences in uncertain databases. *Information Sciences*, 479:76–100, 2019.

[RK13] Margy Ross and Ralph Kimball. *The data warehouse toolkit: the definitive guide to dimensional modeling*. John Wiley & Sons, 2013.

[RN10] Marko A Rodriguez and Peter Neubauer. Constructions from dots and lines. *Bulletin of the American Society for Information Science and Technology*, 36(6):35–41, 2010.

[SAB10] Rania Soussi, Marie-Aude Aufaure, and Hajer Baazaoui. Towards social network extraction using a graph database. In *Proceedings of the Second International Conference on Advances in Databases, Knowledge, and Data Applications*, pages 28–34. IEEE, 2010.

[SBC+17] Neil Swainston, Riza Batista-Navarro, Pablo Carbonell, Paul D Dobson, Mark Dunstan, Adrian J Jervis, Maria Vinaixa, Alan R Williams, Sophia Ananiadou, Jean-Loup Faulon, et al. biochem4j: Integrated and extensible biochemical knowledge through graph databases. *PLOS ONE*, 12(7):e0179130, 2017.

[SdV88] Frans N Stokman and Pieter H de Vries. Structuring knowledge in a graph. In *Human-Computer Interaction: Psychonomic Aspects*, pages 186–206. Springer, 1988.

[Sha18] Ben Sharma. *Architecting data lakes: data management architectures for advanced business use cases*. O'Reilly Media, 2018.

[Shi22] Yong Shi. *Advances in big data analytics: theory, algorithms and practices*. Springer Nature, 2022.

[Sin12] Amit Singhal. Introducing the knowledge graph: things, not strings, Official Google blog, May 16, 2012.

[SL19] Oluwafemi A. Sarumi and Carson K. Leung. Exploiting anti-monotonic constraints in mining palindromic motifs from big genomic data. In *Proceedings of the International Conference on Big Data 2019*, pages 4864–4873. IEEE, 2019.

[Smi93] Herman Jannes Smit. *Consistency and robustness of knowledge graphs*. PhD thesis, University of Twente, Netherlands, 1993.

[SML+21] Wojciech Samek, Grégoire Montavon, Sebastian Lapuschkin, Christopher J Anders, and Klaus-Robert Müller. Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3):247–278, 2021.

[SMÜ19] Rajesh Sharma, Artem Mateush, and Jaan Übi. Tale of three states: Analysis of large person-to-person online financial transactions in three baltic countries. In *Proceedings of the International Conference on Big Data 2019*, pages 1497–1505. IEEE, 2019.

[Sow84] John F Sowa. *Conceptual structures: information processing in mind and machine*. Addison-Wesley Longman Publishing Co., Inc., 1984.

[SS77] John Miles Smith and Diane CP Smith. Database abstractions: Aggregation and generalization. *ACM Transactions on Database Systems (TODS)*, 2(2):105–133, 1977.

[SS98] Pierangela Samarati and Latanya Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, SRI International, Menlo Park, CA, USA, 1998.

[SS23] Sakshi Srivastava and Anil Kumar Singh. Fraud detection in the distributed graph database. *Cluster Computing*, 26:515–537, 2023.

[SW11] Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[Swe02] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[TF76] Robert W Taylor and Randall L Frank. Codasyl data-base management systems. *ACM Computing Surveys (CSUR)*, 8(1):67–103, 1976.

[TKIH19] Shusaku Tsumoto, Tomohiro Kimura, Haruko Iwata, and Shoji Hirano. Estimation of disease code from electronic patient records. In *Proceedings of the International Conference on Big Data 2019*, pages 2698–2707. IEEE, 2019.

[TL76] DC Tsichritzis and Frederick H. Lochovsky. Hierarchical data-base management: A survey. *ACM Computing Surveys (CSUR)*, 8(1):105–123, 1976.

[TSRC15] Ignacio G Terrizzano, Peter M Schwarz, Mary Roth, and John E Colino. Data wrangling: The challenging yourney from the wild to the lake. In *Proceedings of the Conference of Innovative Data System Research (CIDR) 2015*, 2015.

[TYF86] Toby J Teorey, Dongqing Yang, and James P Fry. A logical design methodology for relational databases using the extended entity-

relationship model. *ACM Computing Surveys (CSUR)*, 18(2):197–222, 1986.

[UHC20] Shashanka Ubaru, Lior Horesh, and Guy Cohen. Dynamic graph based epidemiological model for COVID-19 contact tracing data analysis and optimal testing prescription. *arXiv preprint arXiv:2009.04971*, 2020.

[VGO+20] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

[VK14] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[WA15] Coral Walker and Hassan Alrehamy. Personal data lake with data gravity pull. In *Proceedings of the Fifth International Conference on Big Data and Cloud Computing*, pages 160–167. IEEE, 2015.

[WCY+20] Lei Wang, Wei Chen, Wenjia Yang, Fangming Bi, and Fei Richard Yu. A state-of-the-art review on image synthesis with generative adversarial networks. *IEEE Access*, 8:63514–63537, 2020.

[WDA+16] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3(1):1–9, 2016.

[Wes68] Alan F Westin. Privacy and freedom. *Washington and Lee Law Review*, 25(1):166, 1968.

[WFG+24] Xinbing Wang, Luoyi Fu, Xiaoying Gan, Ying Wen, Guanjie Zheng, Jiaxin Ding, Liyao Xiang, Nanyang Ye, Meng Jin, Shiyu Liang, et al.

Acemap: Knowledge discovery through academic graph. *arXiv preprint arXiv:2403.02576*, 2024.

[WSW21] Zhengwei Wang, Qi She, and Tomas E Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[XT06] Xiaokui Xiao and Yufei Tao. Anatomy: Simple and effective privacy preservation. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB 2006)*, pages 139–150, 2006.

[XX15] Yonghui Xiao and Li Xiong. Protecting locations with differential privacy under temporal correlations. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1298–1309, 2015.

[YKK17] Byoung-Ha Yoon, Seon-Kyu Kim, and Seon-Young Kim. Use of graph database for the integration of heterogeneous biological data. *Genomics & informatics*, 15(1):19–27, 2017.

[YKZ20] Radoslav Yoshinov, Igor Kulikov, and Nataly Zhukova. Methods of composing hierarchical knowledge graphs of telecommunication networks. *Problems of Engineering Cybernetics and Robotics*, 72:69–78, 2020.

[ZBM+22] Ariele Zanfei, Bruno M Brentan, Andrea Menapace, Maurizio Righetti, and Manuel Herrera. Graph convolutional recurrent neural networks for water demand forecasting. *Water Resources Research*, 58(7):e2022WR032299, 2022.

[ZXM+10] Yu Zheng, Xing Xie, Wei-Ying Ma, et al. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Engineering Bulletin*, 33(2):32–39, 2010.