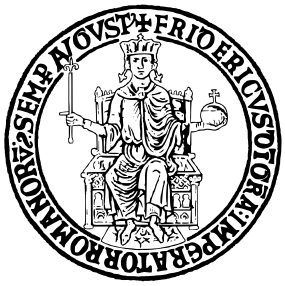


UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

Ph.D
itee
INFORMATION TECHNOLOGY
ELECTRICAL ENGINEERING

Ph.D Thesis in Information Technology and Electrical Engineering

Data Science methodologies for predictive analytics in Smart Cities

XXXIV Ph.D. cycle 2021/2022

Ph.D. Dissertation of:
Vincenzo Schiano Di Cola
matr. DR993631

Supervisors
Prof. Nicola Mazzocca, Prof. Francesco Piccialli
Ph.D School Coordinator
Prof. Daniele Riccio

To you, the reader...

Acknowledgements

This thesis work would not have been feasible without all of the collaborations and professional projects that were available, as well as all of the people I worked with and all of the human support that I received.

Regarding project acknowledgement, the following universities, departments, and research laboratories have all made direct or indirect contributions to the academic development of the thesis:

- the IT company DATABOOZ ITALIA S.r.l., trough Paolo Benedusi,
- the Chung-Ang University in South Korea
- Department of Mathematics and Applications "Renato Caccioppoli"
- Grakn research laboratory,
- the data used in this work were made available through the CETRA (Cultural Equipment with Transmedial Recommendation Analytics) research project, and the "CUP-in-One-Click" research

project.

Finally, the overall thesis is financially supported by the P.O.R. CAMPANIA FSE 2014/2020 ASSE III – OBIETTIVO SPECIFICO 14 Azione 10.4.5

Then I'd like to thank everyone at the MODAL research laboratory for their efforts and support, especially Francesco, Giampaolo, Fabio, Edoardo, Danilo, Andrea, and Ugo.

I'd like to express my gratitude to the administrative staff at the Department of Electrical Engineering and Information Technologies (DIETI) and the professors I met during these three years, particularly Antonio Picariello who passed away too soon.

Finally, I'd want to express my gratitude for Salvatore's pedagogical assistance and Carlo and Rosa's spiritual guidance.

Abstract

The goal of this PhD dissertation is to conduct academic and industrial research on Data Science in a variety of fields. An interdisciplinary approach was required to address today's scientific and societal challenges. A three-year training path applied Data Science to two Smart City application domains: Cultural Heritage (CH) and E-Health, with a focus on machine learning (ML) and knowledge graphs (KG).

The first application is on classifying and forecasting visitor flow within a museum. By applying Machine Learning to the CH sector, the study examined a mixed dataset of numerical and categorical values. A framework for data processing and information extraction for clustering visitor behaviors was developed to save time.

The dissertation then focuses on two e-health topics: healthcare booking prescriptions and image processing for biosensors.

Prescriptions issued by general practitioners were modeled as a KG to help optimize government and local e-health services. This dissertation aimed to identify more patterns in data than a legacy dataset and thus make more accurate predictions.

The final biosensor application recognizes point of interests in smartphone photos and uses machine learning algorithms to determine their chemical composition. The tool predicts the amount of a compound based on the liquid sample's luminescence.

This dissertation's specific research questions concentrate around one question: how can Data Science help construct Smart Cities? This is addressed through a framework for analyzing people moving indoors, an extension of a legacy SQL database to a Knowledge Graph, and the building of a lab-on-hand proof of concept.

All of this is accomplished through the use of a wide range of mathematical and software methods, such as machine learning (clustering and classification), image processing, and KG embedding. Python and R with Grakn, AmpliGraph, OpenCV, and scikit-learn have been utilized as toolkits.

Among the most important contributions made by this thesis are: a data processing framework for clustering visitor behavior (CH domain); tools to help CH decision-makers better analyze visitor behavior and data clusters (both are critical aspects in any kind of ML and decision-making tools); a framework for KG data management and analysis; a framework for biosensors recognizes point-of-interests in smartphone images and uses machine learning algorithms to estimate a compound's concentration in a liquid sample.

Keywords— Data Analytics, Machine Learning, Knowledge Extraction

Contents

List of Figures

List of Tables

Chapter 1

Data Science for Predictive Analytics: An Introduction

The aim of this PhD project was to conduct research in academic and industrial settings and work professionally in multiple domains, by developing predictive models. The aim was to be able to use data-driven decision-making by becoming more technically proficient in the field of data science and meet the demands of the industrial world. This led to the growth of knowledge in all the scientific regarding Data Science, i.e. mathematics, statistics, and computer science. The key issues to be addressed included to be able to use the plethora of information available nowadays in the era of information, the integration of multiple sources of observation like users' social processes or industrial processes, which include: data relating to the fruition and use of goods, services, products, tools, and infrastructure. The research has focused on the application of Data Science methods and techniques to bring industrial innovation to various data types and application domains.

1.1 Industrial Data Science Ph.D. course structure

The overall structure of the Ph.D. was important, as the technology and experiences required for a successful approach to Digital Science and Intelligence were crucial. An interdisciplinary approach had to be used to develop methods and models for interpreting data in response to the challenges of science and contemporary society, using cross-disciplinary skills. This was applied through the joint interest of academic institutions and industrial partners, in particular the University of Naples, a Naples-based IT company DATABOOZ Italia S.r.l. and the Incheon National University in South Korea. Prof. Gwanggil Jeon, from Incheon National University, brought its experience with IoT systems, and image processing. While the Naples-based IT company DATABOOZ Italia S.r.l. specializes in various IT areas, including Business Intelligence and data warehouse, Big Data, Business performance management, with software and process engineering. In particular, they could provide access to a wide range of IT tools, including in-memory, parallel, and distributed processing tools, NoSQL, transactional-analytical databases, Graph Database, and data mining-predictive analytics. Furthermore, the Department of Electronic Engineering and Information Technologies at the University of Naples Federico II plays a pivotal role that is constantly active and open to national and international collaboration aimed at research institutions and businesses. This synergy made it possible to access different data sources, like closed sources, open sources, and sources offered by institutional data providers, and this permitted the implementation of projects focused on attaining short and medium-term objectives and meeting the demands of the industrial world. This system enabled the Ph.D. program to process massive amounts of data in conjunction with machine-generated or human-generated data, all of which needed to be subjected to quality and congruence checks. In fact, predictive analytics takes its power from a wide range

of approaches and techniques, including big data, data mining, statistical modeling, machine learning, and other mathematical techniques.

Also, an interdisciplinary approach had to be used to develop methods and models for interpreting data. Multiple techniques and technologies are utilized to achieve this goal, including knowledge-based techniques such as semantic models and deep learning. Due to the nature of big data, it is incredibly difficult to properly collect social data from different sources. In addition, uncovering patterns in social data to better understand our culture is a difficult task in self.

The development of a framework for data processing and information extraction was critical, as a framework enables for further extension by spending less time on critical tasks (i.e., collecting data, pre-processing data, and analyzing data). Data is also essential in this thesis. The overall project tried to find a narrative into the data. From data collection and storage to analyzing data for data-driven decision making.

1.2 Research output

A number of specific study areas related to the role of the Data Scientist will be examined. This three-year project, in particular, employed Data Science in two application areas for Smart Cities: Cultural Heritage (CH) and E-Health, with a focus on two specific research subjects, machine learning (ML) and Knowledge Graphs (KG).

The overall research output has focused on the use of Machine Learning techniques applied on data coming from the IoT in the cultural heritage domain, and data in the form of a Knowledge graph or images in the domain of e-health.

In particular, the main innovation that will be examined related to the Data Science pipeline, that is bought in this Ph.D. thesis, are:

- the use of Machine Learning techniques for IoT in the Cultural Heritage

domain and unsupervised classification of mixed data: time (numerical) and path (strings);

- the use of KG for modeling general practitioner's prescriptions;
- a tool for biosensor by extracting points of interest in an image and predicting with ML the amount of a compound based on light intensity.

1.2.1 Cultural Heritage

The digital revolution has increased the availability of data that can be used to better understand user behavior and learn about what causes emotion or disappointment during their visit. Each visitor carries an invisible label listing which desires, expectations and needs related to the experience inside a museum, referable to different types of visitors. Museums are undergoing a remarkable transformation, both in terms of internal structure and external relationships. One of the most difficult challenges is accurately classifying and forecasting visitor flow/behavior inside a museum. Data Science played an important role in developing effective data-driven strategies for the valorization and promotion of the Cultural Heritage (CH) domain. Machine Learning approaches can provide new perspectives, allowing knowledge extraction and insights generated from data.

The results are mainly on searching for a suitable number of clusters in the unsupervised learning framework, and explaining he obtained clusters within the museum domain. The data that we worked with came from two Neapolitan museums: the National Archaeological Museum of Naples (MANN) and Castel Nuovo. The MANN, installed IoT sensors so that we were able to track the Bluetooth devices of visitors. The Castel Nuovo recorded the usage of an application installed on a tablet, that visitors could use to enhance their experience in the museum. This thesis presents and discusses the application of a Machine Learning approach to IoT cultural data collected at the National Archaeological Museum of

Naples.

There were actually two projects involved in the gathering of such data. Data gathered for Castel Nuovo was form the M.A.S.T. (Maschio Angioino Smart Tour) project. MAST equipped the museum itinerary of the Castel Nuovo in Naples with a multimedia audio guide based on proximity sensors which tells the artistic heritage present inside the structure in a pro-active way, and gathered data from the visitors interacting with the tablet application while listening to audio scripts, or looking at photos, . The data for MANN came from the Cetra Project which stands for Cultural Equipment with Transmedial Recommendation Analytics and proposes a comprehensive combination of IoT, machine, and Big Data Analytics based on a transmedia concept, with multiplatform storytelling. Visitors' pathways and movements are tracked and integrated using Big Data to better assess actual demand; monitor and measure the efficacy of cultural and tourism efforts, and tailor contents.

The first research goal was to apply Data Science to the CH sector, by extracting knowledge from the available IoT data, BY exploring a kind of dataset composed of mixed data. Eventually, the objective was to propose to CH decision-makers a toolset for better understanding visitors' behaviors.

Among the various papers that are published on such topics, the ones that deal with Machine Learning techniques by giving rise to a consistent research pattern can be considered:

1. "A machine learning approach for IoT cultural data" [104] , DOI: 10.1007/s12652-019-01452-6; analyses the resulting cluster, with comparisons between several similarities and distance functions that suggest us a useful hidden path in the data.

2. "Exploring Unsupervised Learning techniques for the Internet of Things" [23], DOI: 10.1109/TII.2019.2941142, that considers the time feature in order to improve the classification approach and generalize the comparison by

obtaining an ordered F1 Score heatmap representing the comparison among all the experimented clustering methodologies.

3. "Unsupervised learning on multimedia data"[107], DOI: 10.1007/s11042-020-08781-1, discusses the process of obtaining clusters from the Caste Nuovo data. Whose data had more multimedia features, like the number of actions interrupting an audio track or the number of swipes over the available images. For the Castel Nuovo dataset, we try to check whether or not the use of technology can lead to homogeneous behaviors among different visitors interacting with mobile devices.

4. "Decision Making in IoT Environment through Unsupervised Learning"[106], DOI: 10.1109/MIS.2019.2944783, uses the results of the previous papers, to give meaning at the clusters, and propose insights for museum decision-makers. By presenting the MANN case study, which has been extensively studied in Paper 1. and 2., and results on the Castel Nuovo museum of paper 3.

To expedite the activities and keep them focused on the comparison of the two museums, I also used the most recent version of a PowerBI. However, a cumulative animation developed by the IT company in the CETRA project is also available online[1].

All these results are reported in Sections 4.1.1, 5.1.

Finally, my research also did a focus on Knowledge Graphs (KG) from a corporate point for modeling IoT data in the Cultural Heritage domain. I investigated how is to remodel the data as a Knowledge Graph and use such modeling to apply graph techniques, like Node importance and Community detection. Or investigate the use of the vendor for instances in logs, and the correspondence of multiple mac Addresses. The hypothesis is that each museum has its own history and that even

[1]CETRA project website: http://www.cetra-project.it/ and the PowerBI app.

within the same method, there is no general consistency or rule, e.g. among diff rent non-invasive visitor tracking techniques.

After the data is "organized" as a graph using a data modeling tool (e.g. Neo4J, GraphDB, etc). The goal is to embed each node of a graph in a low-dimensional space. In literature there exist many Knowledge Graph Embedding algorithms and selecting the most appropriate for a given dataset is an important challenge. Another challenge is to choose the number of dimensions because given the size of this space, the algorithm will be able to squeeze there more or less information. Moreover, each Knowledge Graph Embedding algorithm has different input parameters to be set up.

A KG graph approach is presented to support museum stakeholder's services through the extraction of knowledge and new insights. The KG represents a unified model for both data of MANN and Castel Nuovo. The framework relies on GRAKN, a novel and intelligent graph database able to model complex datasets, and Amplighrph, a graph embedding system for multiple embeddings.

1.2.2 e-health

E-Health is one of the key sectors in the exponential growth of big data due to four important phenomena: digitalization of diagnostic imaging, digital reporting for paper replacement, the development of biotechnologies, and the explosion of so-called IoMT (Internet of Medical Things). The primary goal in these fields is to retrieve large amounts of data from various sources and add value to this information so that it can be used in more meaningful and intelligent ways.

This thesis presents, in particular, a series of still unpublished results, on the e-health domain, using Knowledge Engineering for prescription annotations, and image processing for the biotechnology aspects.

In particular for the first application domain, Data came from booking data of the Local Health Department of Naples, data of patients booking their health

service with a prescription, referred by their practitioner. Since there were medical prescriptions, this data could be used to predict what kind of health service a patient might be needed, and also predict the load and KPI of health services providers. KG technologies can be used to optimize government and related local e-health services using Machine Learning (ML) and data analytics, or ensure that physicians can retrieve available information in time to assist them in making clinical decisions. We are constantly interacting with KGs on a daily basis, and many research organizations are already utilizing KG methodologies and applications to help them stay ahead of the competition. This thesis presents a KG graph approach to support e-health services through the extraction of medical knowledge and new insights.

I worked on Health care booking data, that used a legacy dataset developed in the 1980s to search insights with AI solutions. The aim of this project was to apply AI solutions on legacy data and generate a KG form that could be used for real-time analytics and prediction. Since the data were not structured into an Entity-relation database and were in a tabular form; an in-depth study of the relationships between pairs of rows sharing the same values on one or more fields has been performed. Moreover, data Science processes, usually require organizing, cleaning the data and structuring the data in a way that can be used by the corresponding algorithm.

Finally, once we obtain this data, then we feed it into our unsupervised and supervised learning algorithms. I investigated how is to remodel the data as a Knowledge Graph and use such modeling to apply graph techniques, like Node importance and Community detection. The purpose is still to map a graph's elements into a low-dimensional space, and the challenges related to KG, like determining the number of dimensions or choosing the best Knowledge Graphs and embedding techniques for a given dataset. The two promising technological solutions so to create this framework, have been: Grakn, a Hyper relation database,

and AmpliGraph for KG embedding, based on Tensorflow. The findings described in this dissertation about KG for prescriptions were presented at the Grakn Cosmos 2020 in London [2].

All these results are reported in Sections 4.1.2, 4.1.3, 5.2, 5.3.

As for the second application domain, a tool for biosensors was developed for chemical-driven predictive analytics applications. By applied machine learning and image, analysis to predict the amount of a specific compound in a sample of liquid based on its luminescence. The framework uses the SWIFT algorithm to recognize points of interest (POI) in smartphone photographs and then use machine learning algorithms (XGBoost, MLP, and Random Forest) to forecast the chemical contents of each POI based on the diffused light intensity data. By detailing mathematically the SWIFT algorithm it would be possible to further extend the recognition algorithm to more machine learning techniques. The framework has been actually used for a biosensor application for a contaminant (mercury) in a sample of water, by recognizing wells in a well plate[102]. These results can be found in Chapter 6.

1.3 Thesis aim and structure

This dissertation's specific research questions concentrate around one question: how can Data Science help construct Smart Cities? This is addressed through a framework for analyzing people moving indoors, an extension of a legacy SQL database to a Knowledge Graph, and the building of a lab-on-hand proof of concept.

All of this is accomplished through the use of a wide range of mathematical and software methods, such as machine learning (clustering and classification), image processing, and KG embedding. Python and R with Grakn, AmpliGraph,

[2]The link to the conference `https://medium.com/vaticle/grakn-cosmos-2020-a-recap-b8ced957dcdc` and the video `https://www.youtube.com/watch?v=BJlZlpBDRTs`

OpenCV, and scikit-learn have been utilized as toolkits.

Among the most important contributions made by this thesis are: a data processing framework for clustering visitor behavior (CH domain); tools to help CH decision-makers better analyze visitor behavior and data clusters (both are critical aspects in any kind of ML and decision-making tools); a framework for biosensors recognizes point-of-interests in smartphone images and uses machine learning algorithms to estimate their chemical composition; and a method for estimating a compound's concentration in a liquid sample.

This dissertation is structured in 7 chapter. This chapter established the context in which this thesis was written and explains the thesis's structure. Chapter 2 defines the fundamental concepts that evolved from the research, most prominently IoT and Knowledge Graph. By posing the conceptual foundations of the thesis, it draws attention to the general literature review and where it has gone and is now. Chapter 3 builds on the previous chapter by delving deeper into the theory of machine learning and technical elements of knowledge graphs and graph embedding. The following Chapter 4 provides describe the predictive approaches used to perform applications, the motivation for choosing one strategy over another, and the tools utilized to collect and interpret the data.

Finally, Chapters 5 and 6 describe the thesis's main findings. Chapter 5 discusses both the results obtained using unsupervised learning in the cultural heritage domain, as well as the machine learning results obtained using KG embedding on -health data from medical prescriptions. Whereas Chapter 6 is a self-contained chapter that discusses the biosensor application framework by describing the technique for extracting certain features from images and the machine learning results for these data features. Chapter 7 summarizes the thesis and suggests additional research challenges.

Chapter 2

About Data Science for Smart Cities

The first chapter of this thesis explains the research being conducted.

It identifies the findings, concepts, debates, and hypotheses that have emerged within the Data Since process. It draws attention to the fact that there are gaps in the literature, whether they are methodological, conceptual, or epistemological. Finally, it poses the foundations of how my thesis fills in the gaps, which serves as a premise for the remaining chapters.

In particular, we will see

1. Problems in determining the optimal number of clusters to use and how distance affects clustering.
2. Optimally improving the outcome of several supervised techniques
3. Open problems in knowledge graphs

2.1 Need for Data Science in Smart Cities

According to the review paper of[134] smart city applications typically have four components: data collection, transmission/reception, storage, and analysis. Input data collecting has been a major motivation for sensor development in numerous sectors. The second phase is the exchange of data, this entails data transmission from the data gathering units towards the cloud for storage and analysis. Many smart city initiatives have city-wide Wi-Fi networks, 4G and 5G technologies are used, as well as various forms of local networks that can transfer data locally or globally. The third stage is cloud storage, where data is organized and stored to be ready for the fourth stage, data analysis. Data Analysis refers to the extraction of patterns and inferences from the obtained data to help decision making. For more complicated decision making, the availability of the cloud allows not only for heterogeneous data gathering/storage and processing but also analysis through the application of statistical approaches as well as machine and Deep Learning algorithms in real-time.

A smart city is made up of numerous areas which are represented in Figure 2.1, in particular, they are:

Smart Farming Precision agriculture is part of the smart agriculture paradigm and involves placing sensors in plants to provide targeted measurements and thus allow for targeted care mechanisms. Data-driven crop care and decision-making are the main applications of AI in IoT for agriculture.

Smart City Services Smart city services include municipal tasks such as water supply, waste management, environmental control, and monitoring. Sensors can also be used to monitor pollution levels in cities and direct citizens to the nearest free parking spot to save fuel.

Smart Energy Because smart grids collect real-time consumer data, they can better manage power generation to ensure uninterrupted supply.

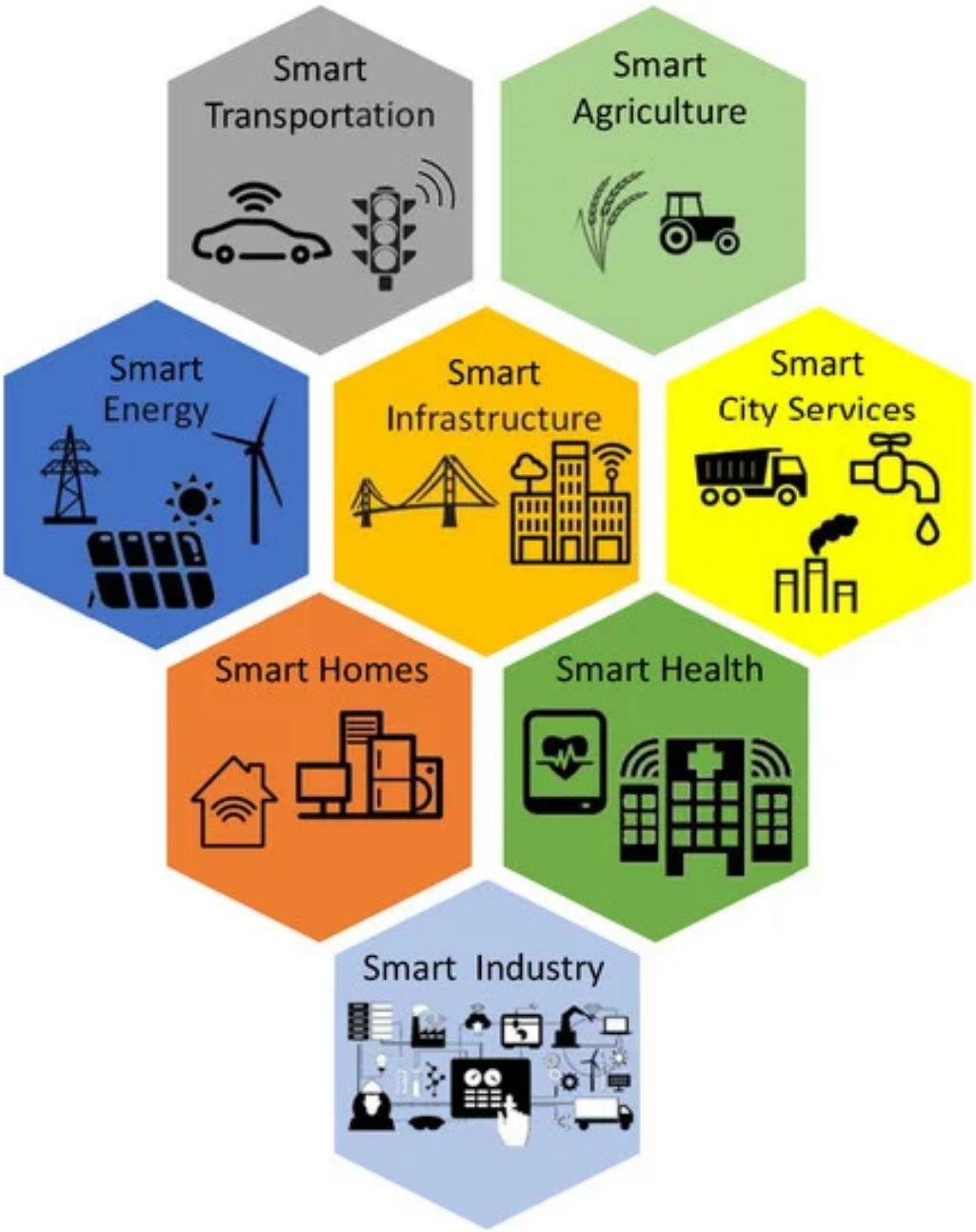


Figure 2.1: Smart City Components: data collection, transmission/reception, storage, and analysis. Credits[134].

Smart Health Smart Health is the use of ICT to improve health care access and quality. These days, with the prevalence of mobile phones and health trackers that can record the daily activity and detect abnormal movements using inertial sensors, it's possible to use cloud processing to make better healthcare decisions.

Smart Home The Smart Home is a major component of Smart Cities because it is the hub of the inhabitants' lives. These sensors may include ambient sensors, motion trackers, and power/energy monitors.

Smart Industry The 4.0 paradigm envisions a connected factory with seamless integration of all intermediary functions. However, working with a heterogeneous set of devices and machines poses unique challenges for IoT applications in Smart Industries. The main industrial applications of AI are predictive maintenance, machine health monitoring, and production management.

Smart Infrastructure To maintain the quality of life in a city, city governments must build new bridges, roads and buildings as well as maintain existing ones. Using accelerometers and smart materials, the smart infrastructure helps cities keep their infrastructure in good shape and usable.

Smart Industry Vehicle-Infrastructure-Pedestrian communication is now commonplace due to rapid technological advancement. This real-time data is already used for route planning in apps like Waze and Google Maps, as well as public transportation. Sensor-equipped parking systems can also direct drivers to nearby free parking.

According to[78] Newcomers of emerging technologies such as the internet of things, cloud computing, big data, or virtual and augmented reality bring new promises and new expectations to data scientists. In the same context, citizens and communities want secure, dependable, trustworthy, high-performing services that

solve problems and add value. According to the current state of Information and Communication Technologies (ICT) evolution, sophisticated information systems incorporate the following features that can be integrated into smart city solutions:

Databases and Data Warehouses technologies In the context of smart city services, Data warehouse technology is a core technology, and data protection, data privacy, and data regulation have all required delicate handling.

Content Management and Collaboration Platforms The abundance of data in structured and unstructured formats complicates the design and delivery of smart city services. Together with Social networks, social media, and collaborative, community-driven technologies are critical components of smart city services.

Advanced Computer Networks technology 5G networks, Internet of Things together with mobile networking and distributed sensor networks all provide added value to for new components in smart cities.

Big Data and Analytics technology this enable the discovery of hidden patterns in large amounts of data and provide policymakers with unique visualization options.

Several highly promising technologies, such as Artificial Intelligence, the Internet of Things, cloud computing, virtual, mixed, and augmented reality, sensors, and 5G networks, have emerged in the last decade. Data Science as an overall paradigm can help to exploit all the promising technologies for the future of smart cities[78].

Moreover,[35], building effective IoT applications for smart environments necessitates the integration of technology, techniques, and skills from a wide range of disciplines, including electronic engineering, data engineering, and data science, as well as user experience design and behavioral science. Figure 2.2 depicts a model for structuring the challenges based on the authors' previous experience developing IoT-enhanced applications for smart environments. The digitization of physical

infrastructure (shown on the left side of Figure 2.2) may generate large amounts of data on resource utilization in a smart environment. It is critical to use a systematic approach to data collection and analysis to manage these resources holistically. The first phase of the model focuses on monitoring and analysis, collecting and analyzing data from the smart environment using IoT platforms and big data processing infrastructure. This data can then be used in advanced decision support tools (for example, predictive and prospective analytics, simulations, and so on) to improve resource performance and optimization in the smart environment.

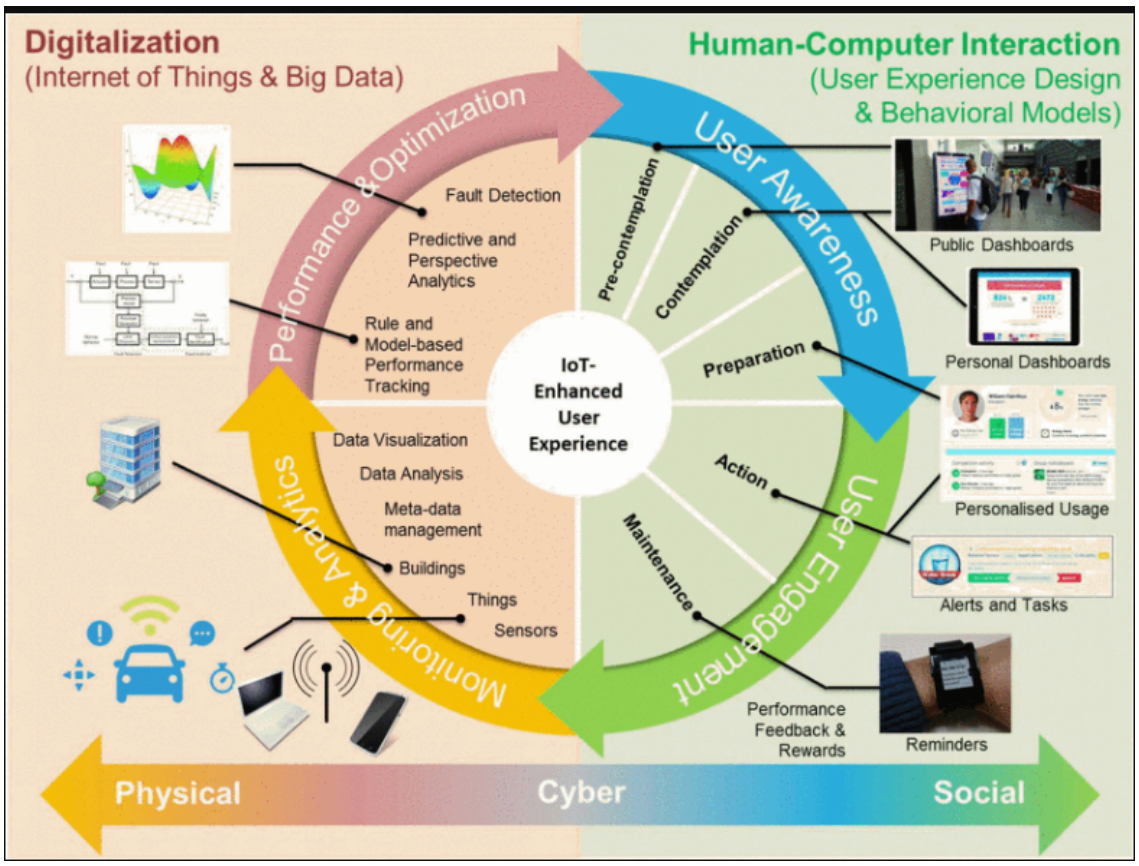
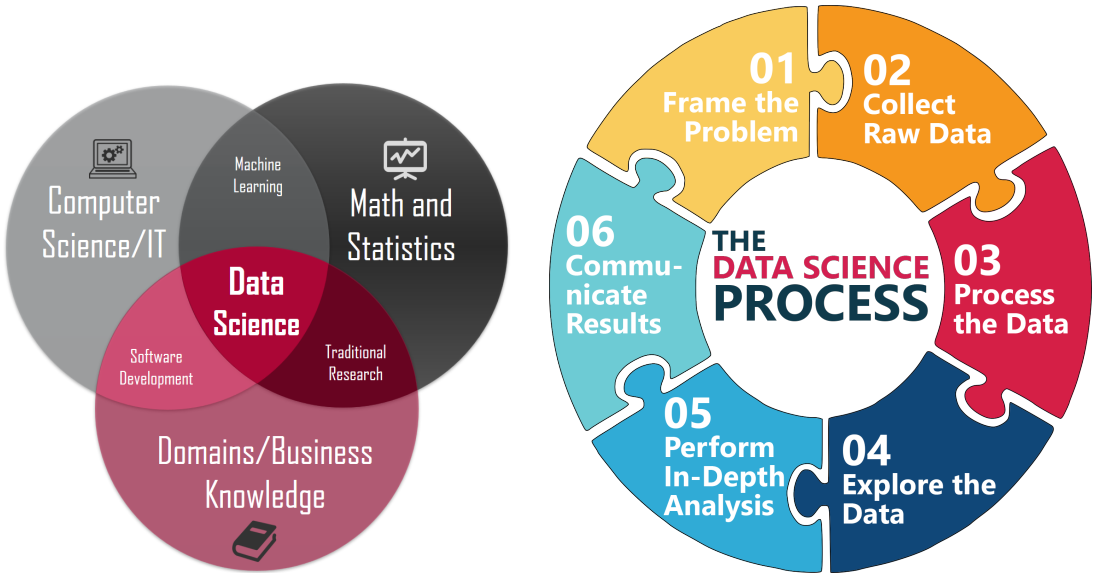


Figure 2.2: A model for IoT-enhanced user experience. Credits[35].

All this needs a general framework to process data, i.e. Data Science. A typical process is depicted in Figure 2.3b. We will investigate such a process according to 3 different application domains: cultural heritage, e-health, and environmental control, with three different data: mobile device generated, human-generated descriptions, and image type.



(a) Data Science (b) Data Science Cirlce Puzzle

Figure 2.3: Data Science Process

2.1.1 Cultural Heritage Domain

The nature of museum visits lends itself to categorizing the patterns that visitors exhibit while walking through it. Museum visitor research dates back to the early twentieth century[84]. These studies may result in a better understanding of visitor needs and the provision of useful services that visitors may desire.

The authors of[41] present a collaborative tagging and navigation system for an online digital museum. Museum visitors can be classified based on the methods they employ: explicit or implicit. Explicit methods are those that use methods that directly determine the categorization, whereas implicit methods are those that use other factors to deduce or imply the categories. Explanatory methods include using questionnaires to determine attitudes and observing behaviors to define movement patterns. Implicit methods include the use of timing and tracking of user movements to determine attitudes or personality traits or watching eye movements to determine whether or not a user liked a particular exhibit.

We can use it to gain a better understanding of the dynamics of the visit and make appropriate changes to the museum.

Emerging innovative technologies are revolutionizing the way museums collect data and enabling us to collect more finely granular space and time resolution [89], starting from the RFIDs [59], Bluetooth [29;149] technologies, or the combination of other wearable devices (for example, see [63]). According to [121], the growing popularity of Internet of Things (IoT) applications has increased demand for the comprehensive exploration of associated IoT design spaces, taking system, network, and human-level perspectives into account.

The collected data are used to classify visitors into pre-established visiting types (i.e., the busy, selective, and greedy) based on their path and length of stops [130], to predict the estimated viewing times and exhibited place to be interested in a visitor based on their previous location and viewed time [15], and to validate the previously made classifications [128;154].

They also allow us to compare mathematical simulations of visit styles [128] with the effects of multi-medial location-aware guides, demonstrating that visitors who use such guides on their mobile devices tend to extend the duration of their visits while decreasing social interactions among members of the same visitor group [95].

In this light, the authors of [82] propose an intriguing study on how visitors distribute their time across artworks in a real case study (the CoBrA Museum of Modern Art); however, the small number of considered volunteered visitors (180) limits the overall considerations and results they propose. However, the gathered datasets were frequently small-scale samples rather than longitudinal, dynamic, or fine-grained data. [150] collected a large-scale sample in a large-scale art museum (i.e., the Louvre museum), revealing underlying patterns of visitors' behaviors in terms of path sequence and length of stay in the museum, but they did not classify the visiting style in terms of visitors' attributions, nor did they use the clustering approach for that purpose.

2.1.2 e-health

Big Data is a buzzword, even in healthcare. However, it is not always clear how scientific data can be used in the medical field. A better understanding of community preferences and needs is the first step, followed by using technology to both improve healthcare and promote advanced research. It represents the challenge of using Big Data in medicine,[110;133].

These days, research projects, publications, and e-health records generate thousands of terabytes of data (EHRs). For example, if carefully analyzed with appropriate Machine Learning (ML) methodologies and tools, these can help researchers and stakeholders in the healthcare domain find new clinical and customized treatments and/or services[159].

In general, healthcare organizations have large databases containing clinical, biological, epidemiological and administrative data. Wearable devices that track mobility, measure biological variables, monitor lifestyles, or provide information on an individual's behavior may also provide medically relevant data. Once Big Data is available, the research challenge is how to use it, which hides a more important issue, how to learn from it. There are personal data about the patient and administrative data about the healthcare management authority.

One of the main disadvantages in this scenario is the lack of efficient digital e-health data for the learning algorithms. A large amount of data does not automatically equate to better inferences and applications. The data are useless. Data must be selected, structured, and interpreted to be useful[148]. Thus, it is not the technology but the ability to extract value from it that matters. A social construct, data are the result of specific cultural, social, technical and economic choices made by individuals, institutions, or companies for the collection, analysis, and use of data. The data from the "real world", such as electronic registers or medical records, are not only unstructured but also limited in scope.

Big Data is no longer a buzzword. The knowledge contained in data is represented by its value in computer science and learning technologies. Initially, descriptive analytics software could display data in simple graphs and tables to describe phenomena. Then came predictive analytics software, which used algorithms to identify data correlations and link them to specific situations. A promising and challenging research area in recent years, especially in the e-health domain[114]. KG is a technology for storing and managing data graphically. It can plot the relationships between any of its data points. For example, unsupervised and supervised learning algorithms can benefit from KG's structured representation of real-world noisy data.

To meet the growing demand for medical decision support systems (DSS), the authors of[114] proposed a learning approach based on existing disease-symptom links. Notably, EHRs are the primary source of digital patient health data containing medical knowledge[118;135]. Using Chinese EHRs, the authors of[56] proposed a method for creating a medical diagnosis knowledge network (MKN). In[124], the authors propose a KG approach through a clinical Bayesian network. The authors of[80] proposed an automated KG construction related to cerebral aneurysms disease.

As shown in Section 3.2, most KG research studies are recent, encouraging us to investigate and contribute to this research field. Based on the above, this thesis focuses on the difficult task of extracting meaningful insights from data using KG. A data graph model was created using a large amount of raw healthcare administrative data. In this age of knowledge as the new gold, the main goal is to design a relation-based data representation and then apply learning techniques to provide more intelligent insights and services to healthcare stakeholders. Technically, we use GRAKN (http://grakn.ai), a novel and intelligent graph database that can model complex datasets.

This section's objective is to report the related works. This thesis organizes

knowledge from a legacy database, by modelings entity relation in a KG and applies machine learning methods to show new viewpoints on such restructured data. So this thesis encompasses a few areas of research, and a review, to the best of the knowledge of the authors, is addressed in this section. First of all, how to extract knowledge from an uncleaned structured data set, then model and save such knowledge in a graph database, and finally, apply machine learning to show insights. All these areas are linked to either the technological advances or their application to the health care systems and booking applications.

2.1.3 Iot and Knowledge Graphs

The Internet of Things (IoT) connects various physical objects with ubiquitous intelligence and pervasive interconnections. IoT devices with varying functions are typically manufactured by different companies, resulting in varying access standards, demonstrating significant heterogeneity in terms of access methods. An IoT application's main challenge is to determine how different IoT devices communicate effectively in a complex system IoT middleware (Figure 2.4), a software system layer designed to act as an intermediary between IoT devices and applications, is a critical technology for seamlessly integrating different IoT devices into an IoT system. However, two issues, communication gaps, and heterogeneous access prevent existing IoT middleware from being used effectively in an IoT system with disparate standards and interfaces. According to[146], they propose a knowledge graph-based multilayer IoT middleware to address this problem, inspired by a graph-based knowledge system for eliminating heterogeneity in business systems. The proposed multilayer IoT middleware adds a new layer to bridge the communication protocol gap between IoT devices. It can manage all IoT devices uniformly by utilizing an IoT knowledge graph.

According to[8] the primary technical issues with smart city solutions concern data collection, aggregation, reasoning, data analytics, access, and service delivery

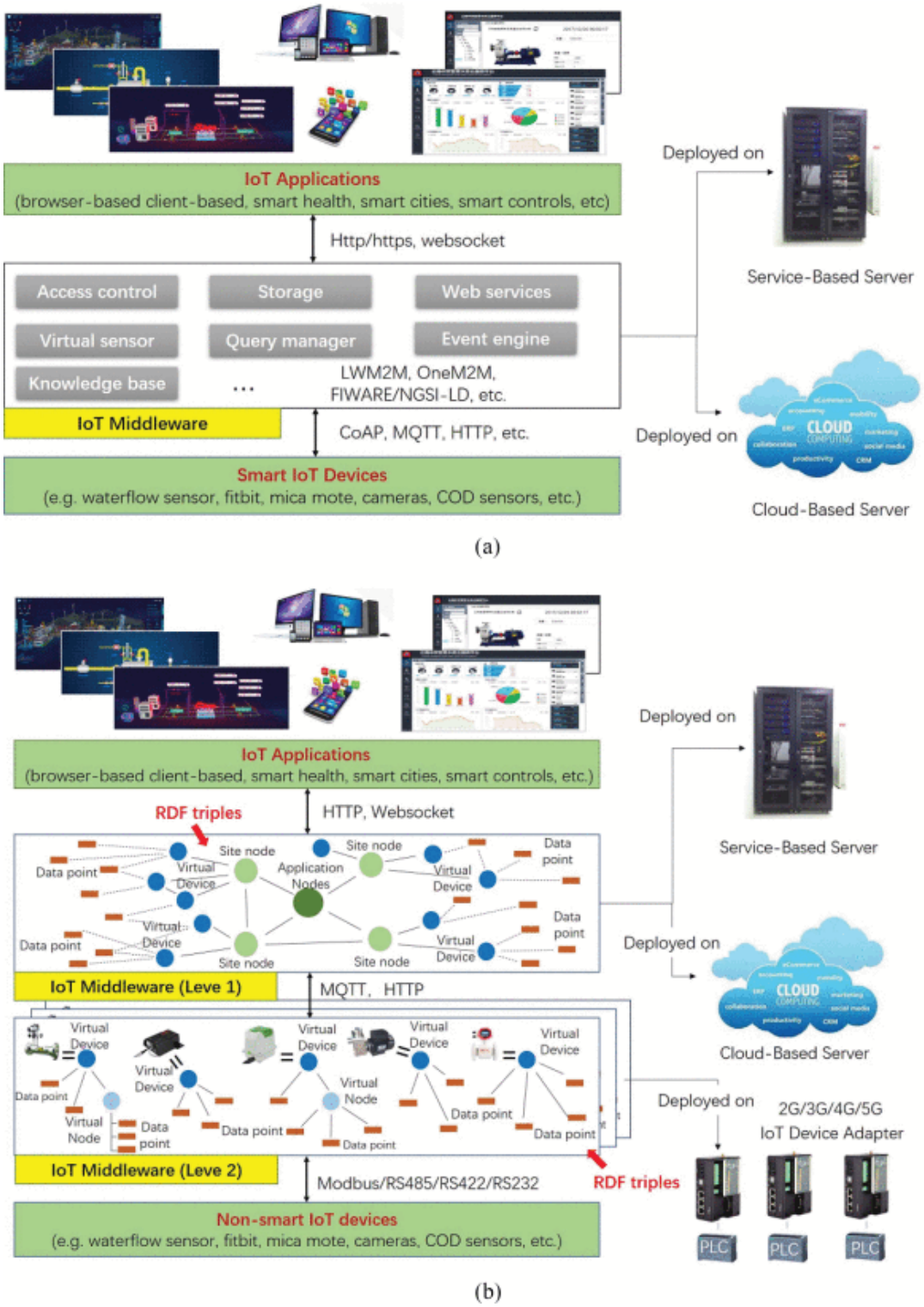


Figure 2.4: (a)Traditional IoT middleware for industrial applications utilizing smart IoT devices. (b) Knowledge graph-based IoT middleware for non-smart IoT devices in industrial applications. Credits[146].

via Smart City APIs (Application Program Interfaces). Smart City APIs of various types enable smart city services and applications, while their effectiveness is determined by architectural solutions for passing data from data to services for city users and operators, utilizing data analytics, and presenting services via APIs. As a result, there is a lot of work being done to define smart city architectures to deal with this complexity, putting in place a wide range of different kinds of services and processes.

[8] presents a comparison of state-of-the-art smart city architecture solutions for data aggregation and Smart City API, demonstrating the use of semantic ontologies and knowledge bases in data aggregation in the production of smart services (Figure 2.5). The proposed the Sii-Mobility smart city project on defining a smart city architecture addressing a wide range of processes that aggregates and re-conciliates data (open and private, static and real-time) through the use of reasoning/smart algorithms to enable sophisticated service delivery via Smart City API.

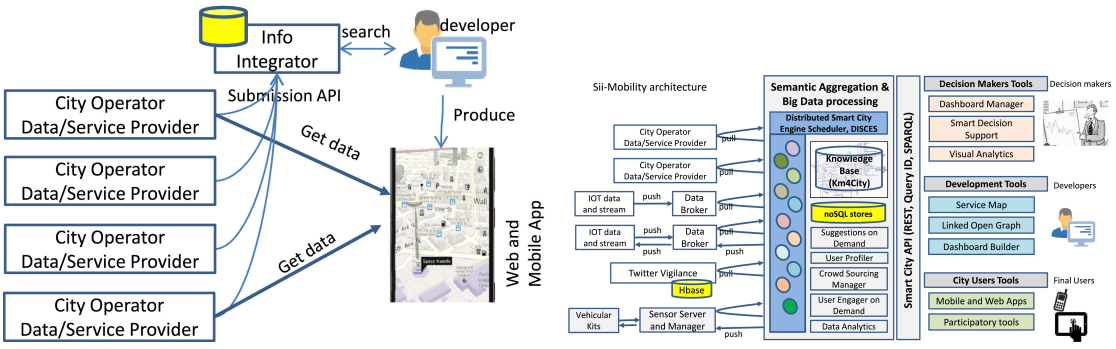


Figure 2.5: From data to services in smart city architectures. Data streams and data services are carried by the thicker lines. For case (a) it integrates only informtions. For case (b) there is the semantic aggregator and reasoner within the Sii-Mobility smart city architecture Credits[8].

According to Gartner in 2020,[46], despite the global impact of COVID-19, 47% of artificial intelligence (AI) investments were unchanged since the start of the pandemic and 30% of organizations planned to increase such investments, according

to a Gartner poll. Only 16% had temporarily suspended AI investments, and just 7% had decreased them. Moreover, in Gartner Artificial Intelligence Hype Cycle for 2021[47], organizations are rapidly embracing AI solutions to generate new goods, improve existing products, and grow their client base using NLP and upcoming technologies like generative AI, knowledge graphs, and composite AI. But less human decision making, more AI decision-making magnify both the positive and bad aspects. Unchecked, AI-based approaches can perpetuate prejudice, causing difficulties, productivity loss, and financial losses. Contrary to popular belief, algorithms cannot detect unconscious bias. For example, a data scientist may fail to see that the number of clicks on a website can be age discriminating. AI can correctly describe a stereotypical Western wedding but fails to recognize weddings in India and Africa. Leaders in D&A and IT are now focusing on "small data" and "wide data" analytics strategies. To use data more efficiently, they can either work with smaller data sets or extract more value from unstructured data sources.

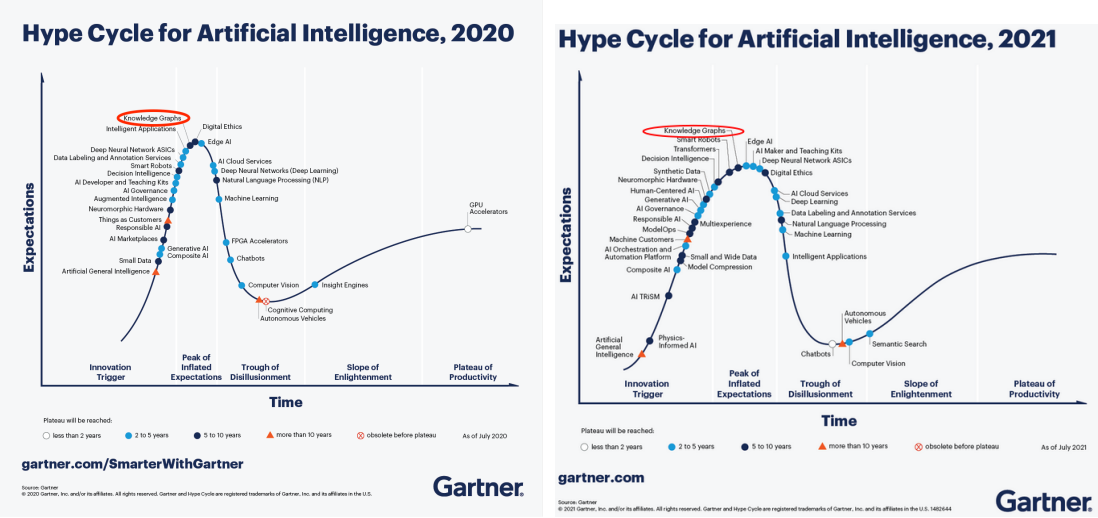


Figure 2.6: According to both Gartner's 2020 and 2021 Hype Cycle for Artificial Intelligence, Knowledge Graphs, have and continue to be on the Peak of the Inflated Expectations, with a 5 to 10 years, before reaching the plateau. Credits[46;47].

2.1.4 Knowledge extraction

The goal of information extraction is to process unstructured information sources and extract a specific kind of information into a structured repository such as a relational table[117]. Sub-tasks of information extraction[57] includes: entity recognition, relation extraction[126] and event extraction[51;155].

To handle big data, researchers have been proposing a range of solutions for entity recognition,[28], most have in common a sequence of steps, that start from blocking procedures to reduce the number of comparisons, and then matching by searching for similarity functions so to minimize the number of false-positive or negative matches. In[97], authors examine the literature of two different but related frameworks, Blocking (a.k.a. indexing) and Filtering, for Entity Resolution. A core task of Data Integration that detect different entity profiles that correspond to the same real-world object. String similarity, on the other hand, is still an important operation for join operations and entity resolution[151],[91].

Although many papers cope with the issue of extracting information from text documents, a still challenging task is to recognize entities and relations from tabular data. This problem is also known as KG identification (KGI), i.e. exacting entities, attributes, and relations among those entities. An example to be mentioned in the recent Oxford *Semantic Web Challenge on Tabular Data to Knowledge Graph Matching*[2]. In this research area, a probabilistic soft logic approach for KGI has been presented in[108]. The authors in[70] also discuss a recent approach based on *retrofitting with functional relations* applied on large healthcare ontologies. An exemplary paper instance on this topic area, considering a probabilistic soft logic approach for KGI, has been presented in[108], or another recent approach is *retrofitting with functional relations* in[70] applied, also, on large healthcare ontologies. These methodologies try either to match new data to a known ontology or to look at embedding to determine entity relations. Some recent works have focused on building up new models in which new knowledge is inferred from com-

puting co-occurrences of concepts[43], measuring the distance between concepts[127], or processing natural language sentences to incorporate the belief state of the physician for assertions in the medical record[115]. Due to the increasing demand for decision support systems (DSS) in medicine, the authors in[114] proposed a learning approach to infer knowledge from existing disease-symptoms links.

2.1.5 Challenges in Industry for KGs

According to[94] and and graknlabs developers[45]. there is a wide range of challenges that still need to be addressed, like completing a KG, Making Decisions, and predicting Relations by forecasting binary, ternary, or N-ary relationships.

Entity disambiguation and managing identity While entity disambiguation and resolution has been an active research area in the semantic web and now knowledge graphs for several years, it is surprising that it remains a top industry challenge. The challenge is to give mention of an entity a normalized identity and a type. Many automatically extracted entities have very similar surface forms, such as people with the same or similar names, movies, songs, and books. Product names can refer to different listings. Incorrect linking and disambiguation will lead to incorrect inferences downstream.

While these issues may be obvious in smaller systems, they become much more difficult when dealing with a diverse contributor base at scale. How can identity be described so that different teams can agree and understand each other? How can developers ensure they have enough human-readable data to resolve issues?

Type membership and resolution Most current knowledge-graph systems allow each entity to have multiple types, each with different implications. For example, Barack Obama is a person, but also a politician and an actor, albeit a less well-known one. Cuba is a country or its government. The type assignment

	Data model	Size of the graph	Development stage
Microsoft	The types of entities, relations, and attributes in the graph are defined in an ontology.	~2 billion primary entities, ~55 billion facts	Actively used in products
Google	Strongly typed entities, relations with domain and range inference	1 billion entities, 70 billion assertions	Actively used in products
Facebook	All of the attributes and relations are structured and strongly typed, and optionally indexed to enable efficient retrieval, search, and traversal.	~50 million primary entities, ~500 million assertions	Actively used in products
eBay	Entities and relation, well-structured and strongly typed	Expect around 100 million products, >1 billion triples	Early stages of development and deployment
IBM	Entities and relations with evidence information associated with them.	Various sizes. Proven on scales documents >100 million, relationships >5 billion, entities >100 million	Actively used in products and by clients

Figure 2.7: Common characteristics of the knowledge graphs at industry-scale as for[94].

is delayed in some knowledge-graph systems: Depending on the user task, the application uses a specific type and collection of attributes.

Initially, enforcing class membership criteria while maintaining semantic stability may be simple. In Google's knowledge graph, e-sports were not defined as a category. So how does Google keep the sports category identity while including e-sports?

Knowledge-graph semantic embeddings. With a large-scale knowledge graph, developers can build high-dimensional representations of entities and relations. The resulting embeddings will greatly benefit many machine-learning, NLP, and AI tasks as sources of features and constraints, and can form the basis for more sophisticated inferences and ways to curate training data. Deep-learning techniques can be applied to problems of entity deduplication and attribute inference.

Knowledge inference and verification Making sure facts are correct is critical in constructing a knowledge graph, and manually verifying everything is impossible. Automated consistency checking and fact-checking can be achieved using advances in knowledge representation and reasoning, probabilistic graphical models, and natural language inferences.

Federation of global, domain-specific, and customer-specific knowledge Clients of IBM who create their custom knowledge graphs are not expected to provide basic knowledge to the graph. A cancer researcher will not teach the knowledge graph that skin is a type of tissue or that St. Jude is a Memphis, Tennessee hospital. This is a general knowledge graph.

For example, carcinoma is a type of cancer, and NHL stands for non-Hodgkin lymphoma rather than National Hockey League (though it can still mean that in some contexts, like a player's medical record). The client should only need to input private or confidential information that the system does not already have. This requirement raises issues of isolation, federation, and online updates of the base and domain layers.

Security and privacy for personalized, on-device knowledge graphs Because knowledge graphs aim to create an entity for every noun in the world, they can only be run in the cloud. In reality, most people only care about a small subset of the world's entities that are personally relevant to them. Personalizing knowledge graphs for individual users holds a lot of promise, possibly even allowing them to be shipped to mobile devices. More on-device learning and computation over local small knowledge-graph instances will allow developers to provide user value while maintaining privacy.

Multilingual knowledge systems A comprehensive knowledge graph must cover facts expressed in multiple languages and conflate the concepts expressed

in those languages into a cohesive set. In addition to the challenges in knowledge extraction from multilingual sources, different cultures may conceptualize the world in subtly different ways, which poses challenges in the design of the ontology as well.

Managing changing knowledge. An effective entity-linking system must also evolve organically with its input data. Companies may merge or split, or new scientific discoveries may split an existing entity. Does a company that acquires another company change its name? What about a division? Does acquiring a name confer identity?

While most knowledge-graph frameworks are becoming more efficient at storing a point-in-time version of a knowledge graph and managing instantaneous changes to knowledge graphs to evolve the graph, there is a gap in the ability to manage highly dynamic knowledge in the graphs. To capture these changes, a fundamental understanding of temporal constructs, history, and change with history is required. Furthermore, the ability to manage updates via multiple stores (such as IBM's polymorphic stores) is required.

The integrity of the update process, eventual consistency, conflicting updates, and runtime performance are all factors to consider. Existing distributed data stores can be modified to handle incremental cascade updates. It's also critical to managing changing schemas and type systems without breaking existing knowledge. Google, for example, divides the metamodel layer into multiple layers. Meta types (which are instances of types) are used to enrich the type system and build higher levels.

Knowledge extraction from multiple structured and unstructured sources. Despite recent advances in natural language understanding, extracting structured knowledge (entities, types, attributes, and relationships) remains a challenge. Scaling up graphs requires unsupervised and semi-supervised

knowledge extraction from unstructured data in open domains.

Examples of graph relationships extracted from the unstructured text are found in eBay product listings and seller catalogs, while the IBM Discovery knowledge graph relies on documents to support its facts. Annotations by humans are required to train traditional supervised machine learning frameworks. Unsupervised (clustering with vector representations) or semi-supervised techniques can reduce or eliminate this high cost (distant supervision with existing knowledge, multi-instance learning, active learning, and so on). Unstructured text can be linked to entities in the graph using entity recognition, classification, text, and entity embeddings.

Managing operations at scale. Not surprisingly, all of the knowledge-graph systems described here struggle with scalability. This dimension often makes problems that have been addressed in multiple forms in academia and research present new challenges in the industry. Managing scale is a fundamental issue that impacts performance and workload directly. Management of fast incremental updates to large-scale knowledge graphs, as at IBM, or managing consistency on a large evolving knowledge graph, as at Google.

Chapter 3

Predictive framework

This chapter serves as the theoretical foundation for my thesis, in particular, will go through the theoretical framework that was used to perform the research. The elements that inspired my choice of these theory subsets, as well as the implications of adopting such a model. Examples of how others have used the proposed theoretical framework in similar settings are offered. It will focus on theoretical ideas such as clustering, supervised learning, and Knowledge Graph embedding.

3.1 Machine Learning

Machine learning (ML) is the study of statistical models and algorithms that use variables from a dataset to recognize meaningful patterns. ML empowers computers to make rapid predictions based on new data without explicit instruction [85]. As systematized in [7] the steps for using a machine learning algorithm (MLA) are as follows:

1. Data Collection. Before building the model, gather appropriate historical, experimental, observational, or simulation-derived data.

2. Data Preparation. It involves data normalization, scaling, and randomization, which is necessary since data variable ranges might affect MLA performance.

3. Data Exploration and Visualization. To avoid biasing the generated model towards predicting a specific range of variables, data exploration and visualization are essential.

4. Data Pre-Processing. It splits data into three components. A model's performance is tested or evaluated during training and then validated or fine-tuned MLA hyperparameters to further increase model performance.

5. Predictions. These are finally made by proving the value of machine learning.

ML differ in their approach, variables handled, and jobs solved. They fall into three categories: supervised, unsupervised, and reinforced[14] (see Fig. 3.1), each with its own advantages and disadvantages. Supervised learning uses a large training dataset of input-output pairs to construct a model that links input and output variables. For classification and regression the techniques include MLR, ANN, SVM, DT, Naive Bayes (NB), and Random Forest (RF). Unsupervised learning detects unknown patterns in a dataset to recognize target variables without the output data. Unsupervised learning algorithms for clustering, feature selection, and dimensionality reduction include Hierarchical Clustering, K-means, and Principal Component Analysis (PCA). Reinforcement models takes an action as input and returns a maximum predicted reward as output, influenced by environmental feedback. Reinforcement learning is explored in various fields, including operations research and Gaussian Process.

3.1.1 Unsupervised Learning

Categorical data are fields that cannot be naturally ordered in the same way that numerical values can, because distance and similarity functions are not naturally

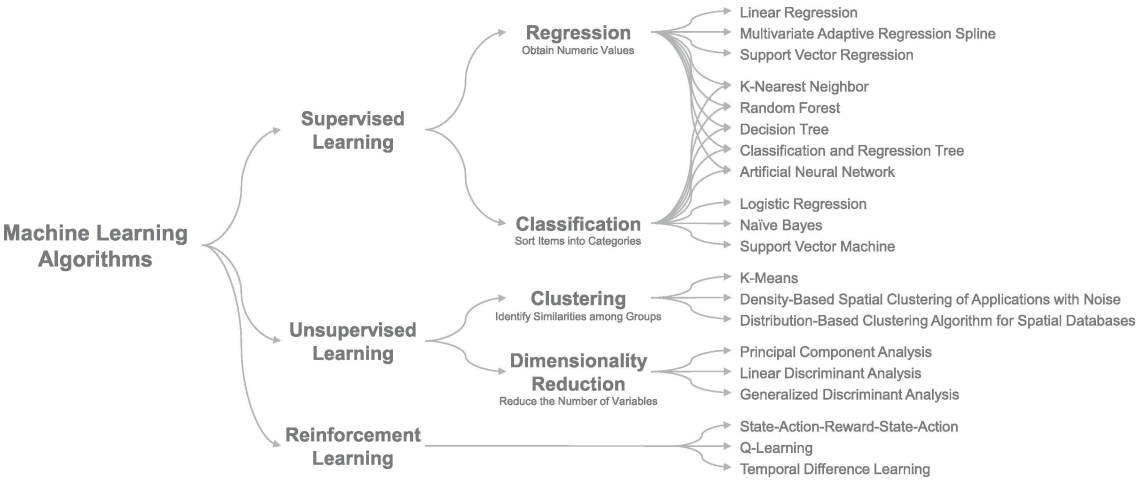


Figure 3.1: Machine Learning algorithms classified according to their learning technique. Credits[7].

defined[16]. The problem of clustering and classifying categorical data is complicated because it involves determining an appropriate distance metric between data[39].

From similarity to distance function

From similarity to distance function:

$$\delta\left(\mathbf{x}_1, \mathbf{x}_2\right) = 1 - s\left(\mathbf{x}_1, \mathbf{x}_2\right)$$

To deal with the numerical *Time* feature we have pre-computed the distance matrix by using the well-known *Euclidean distance* and then merged with the other strings' distances with the formula defined as follows:

$$D_{sum} = \sum_{i=1}^{T} \omega_i D^{(i)}$$

where $D^{(i)}$ is a single distance matrix, T is the total number of distances that we want to merge and ω_i is the inverse of the maximum element of the i-th distance matrix $D^{(i)}$. It's shown that, to improve the performance of a classification, it is better to combine multiple distances than to use a single distance matrix.

Passing from a similarity to a distance is uncomplicated, thus weather we have a sum or a distance, is no problem to us.

To handle the numerical Time feature, we pre-computed the distance matrix using the well-known Euclidean distance and then merged it with the distances of the other strings using the following formula:

It is demonstrated that combining various distances rather than using a single distance matrix improves classification results.[52].

Inertia Criterions The property that two individuals in the same cluster are closer than any other individual in another cluster is too strong.

Within-cluster inertia criterion: we select the partition $\mathbf{z}$ *minimizing* the criterion

$$W(\mathbf{z}) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \delta(\mathbf{x}_i, \overline{\mathbf{x}}_k)^2$$

Between-cluster inertia criterion: we select the partition $\mathbf{z}$ *maximizing* the criterion

$$B(\mathbf{z}) = \sum_{k=1}^{K} n_k \delta(\overline{\mathbf{x}}, \overline{\mathbf{x}}_k)^2$$

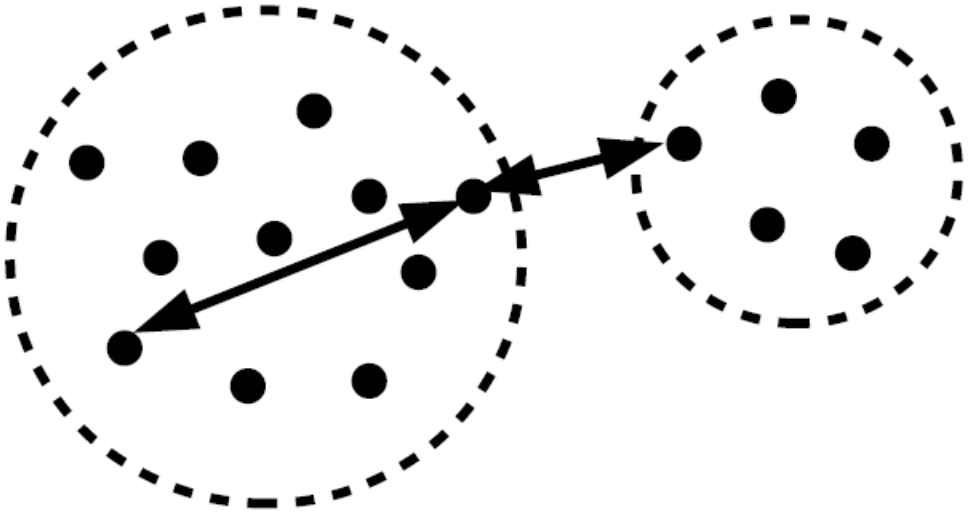


Figure 3.2: Inertia intra and inter-clusters criteria

Once we have a final distance and thus a clustering space, we must address a

problem, to determine the partition's quality. When is a partition an appropriate choice? If individuals are assigned to the same class, they are close to one another. If individuals are assigned to separate classes, they are far apart. Mathematically speaking, there is little within-class variability. There are two criteria: minimum and maximum. Intraclass inertia is a metric that indicates the compactness of each class (cluster). Intraclass inertia is calculated by adding the average squared distances between each element and the class centroid. The partition's between-cluster inertia represents the separation of the clusters. It is the sum of all the distances between the data center and each centroid.

Silhouette index Given:

- Cohesion $a(\mathbf{x}_i) = \frac{1}{|z_i|-1} \sum_{j \in z_i, i \neq j} \delta(\mathbf{x}_i, \mathbf{x}_j)$
- Separation $b(\mathbf{x}_i) = \min_{k \neq i} \frac{1}{|z_k|} \sum_{j \in z_k} \delta(\mathbf{x}_i, \mathbf{x}_j)$

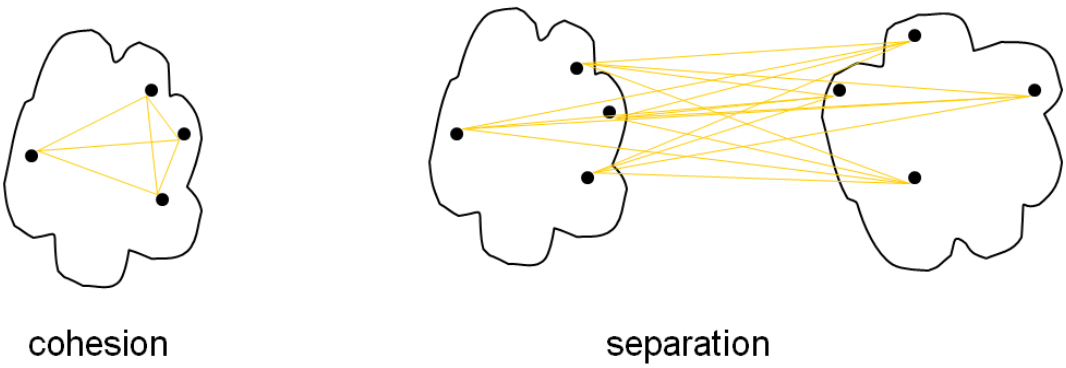


Figure 3.3: Cohesion and separartion.

The silhouette is defined as:

$$s(i) = \frac{b(\mathbf{x}_i) - a(\mathbf{x}_i)}{\max\{a(\mathbf{x}_i), b(\mathbf{x}_i)\}}$$

Cohesion a(i) is a measure of how dissimilar to its own cluster, a small value means it is well matched. average distance between element i and all the other

points in its own cluster average distance of element x to all other vecotrs in the same cluster

Separation distance between element i and its next nearest cluster centroid. Average distance of element x to the vectors in other clusters. Ind minimum among the clusters. Large b(i) implies that i is badly matched to its neighbouring cluster.

So s(i) over all points of a cluster is a measure of how tightly grouped all the points in the cluster are. Thus the average s(i) over all data of the entire dataset is a measure of how appropriately the data have been clustered. s(i) is close to negative one, then by the same logic we see that i would be more appropriate if it was clustered in its neighbouring cluster. An s(i) near zero means that the data is on the border of two natural clusters.

Thus silhouette plots and averages may be used to determine the natural number of clusters within a dataset.

3.1.2 Similarities and distance functions

This section is about similarity and distance functions. More in-depth, in order to analyze unstructured data it is very useful to define a suitable way to measure distances among data and how to *aggregate* the information in homogeneous clouds. The performance of a clustering approach strongly depends on the adopted similarity function and also by the choice of a distance function over the collected data. A similarity function is an instrument that is able to evaluate the strength of the relationship between data items, while dissimilarity deals with the divergence between data. For these reasons, a recall of similarity and distance functions used is needed.

Considering that our dataset is composed by strings representing paths, in order to look at the problem by different perspectives we used either edit-based distances (generalized Levenshtein, Longest Common Substring), and q-gram based distances

(Jaccard and cosine). The q-grams based distances are calculated from q-grams based similarities with the formula $d_x(S,T) = 1 - s_x(S,T)$ as explained in[24].
A string can be defined as a sequence of finite characters from a finite alphabet. We denote a finite alphabet as A and the number of elements in A as $|A|$. The set of all finite strings is A^*. A q-gram is a string with q consecutive characters.

Definition 3.1.1. Let S, T be two strings on A^* of length n, m respectively. Let $v(S;q)$ be a nonnegative integer vector of dimension $|A|^q$ whose components represent the number of occurrences of every possible q-gram in S. The *cosine similarity* between S and T is:

$$s_{cos}(S,T;q) = \frac{v(S;q) \cdot v(T;q)}{\|v(S;q)\|_2 \, \|v(T;q)\|_2}, \tag{3.1}$$

where $\|\cdot\|_2$ indicates the standard Euclidean norm.

In our case, since strings are represented as vectors, the cosine similarity is able to measure the proximity of two strings by looking at the angle between their vector representation. This similarity can be obtained by calculating the cosine of the angle between the two string vectors. Vectors $v(\cdot\,;q)$, counting the occurrences of q-grams, are always non-negative. This means that the cosine ranges from 0 to 1.

Definition 3.1.2. Let S, T be two strings on A^* of length n, m respectively. Let $Q(S;q)$ be the unique set of q-grams occurring in S. The *Jaccard similarity* between S and T is:

$$s_{\text{Jac}}(S,T;q) = \frac{|Q(S;q) \cap Q(T;q)|}{|Q(S;q) \cup Q(T;q)|}, \tag{3.2}$$

where $|\cdot|$ indicate set cardinality.

The Jaccard similarity represents an useful string matching index. It compares elements of two strings in order to find which q-gram are shared and which are not.

The higher the value the more similar the two strings are. This function results extremely sensitive to samples of small sizes and missing observations.
The main difference between the cosine similarity to the Jaccard similarity is that the latter doesn't depend on the occurrence of q-grams.

Definition 3.1.3. Let S, T be two strings on A^* of length n, m respectively. The *Longest Common Substring* distance d_{LCS} is defined as

$$LCS(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ LCS(i-1,j-1) & \text{if } S(i) = T(j), \\ 1 + \min\{LCS(i-1,j), LCS(i,j-1)\} & \textit{otherwise.} \end{cases} \tag{3.3}$$

where $LCS(i,j)$ is the distance between the first i characters of S and the first j characters of T.

The d_{LCS} between two strings p and q is the minimal number of insertions and deletions needed to transform p into q, returning the length of the longest sub-string that p and q have in common. In the case of two strings that are entirely different its value is zero.

Definition 3.1.4. Let S, T be two strings on A^* of length n, m respectively. The *generalized Levenshtein distance* d_{Lv} is equal to the edit distance where all cost functions are equal to 1, and is given by

$$Lv(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} Lv(i-1,j) + 1 \\ Lv(i,j-1) + 1 \\ Lv(i-1,j-1) + 1_{(S_i \neq T_j)} \end{cases} & \text{otherwise.} \end{cases} \tag{3.4}$$

where $1_{(S_i \neq T_j)}$ is the indicator function whose value is 0 when $S_i = T_j$ and 1 otherwise, and $Lv(i,j)$ is the distance between the first i characters of S and the first j characters of T.

The Levenshtein distance between two strings p and q is similar to the LCS distance considering also substitutions to transform p into q. This function is quite expensive in space and time computational complexity when the strings are long. It requires $O(mn)$ operations, where m and n are the lengths of p and q, respectively.

In this[104] we explore the non-normalized version as in (3.3) and (3.4) instead of the normalized one $\frac{LCS(n,m)}{n+m}$ and $\frac{Lv(n,m)}{\max\{n,m\}}$. This choice is made to give insights on the exact number of edit operations.

In the experimental scenario the overall calculations have been implemented in the R framework with the `stringdistmatrix` package, which computes pairwise string distances (see[138]). The distance functions have been implemented as a C library to improve the performance.

All the discussed four similarities and distance functions will be analyzed in the context of clustering in the next chapter.

3.1.3 Clustering and Similarity Metrics

Clustering is the process of locating a subset of data. This example is similar to the standard partition definition. The fundamental aim is to group similar objects with other similar objects.

Data set of n individuals $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, $\mathbf{x}_i$ described by d variables

Clusters:

a set of K non-empty parts of $\mathbf{x} : P = (G_1, \ldots, G_K)$:

- For all $k \neq k', G_k \cap G_{k'} = \emptyset$
- $G_1 \cup \ldots \cup G_K = \mathbf{x}$

$\mathbf{x} = \{a, b, c, d, e\}$

$$P = \{\underbrace{\{a, b\}}_{G_1}, \underbrace{\{c, d, e\}}_{G_2}\}$$

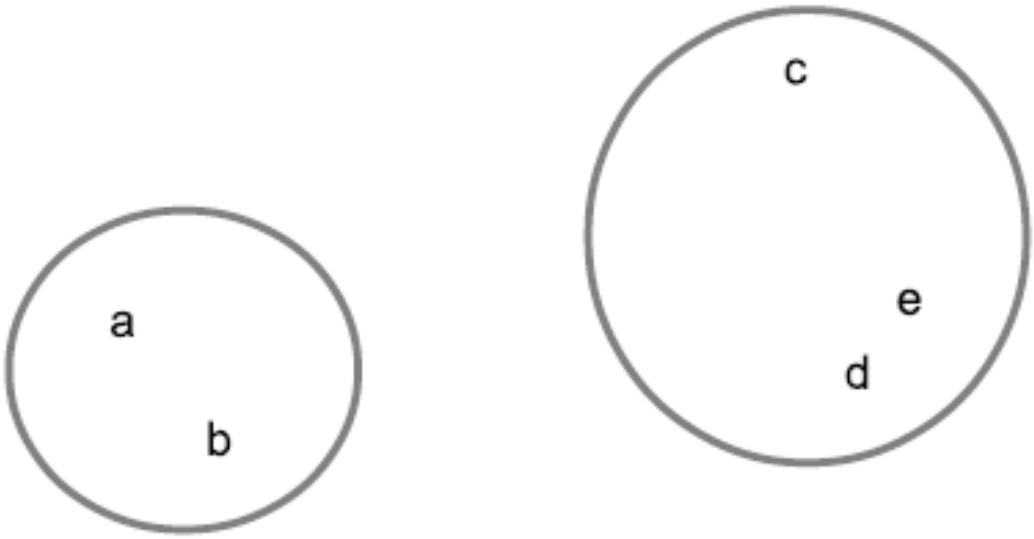


Figure 3.4: Clustering example.

Similarity and distance The first properties of a distance are provided by dissimilarity. On real-world data, such distances are known as the Euclidean distance. Our issue, though, was to find similarities between paths.

We investigated four string similarity methods to cope with our dataset's non-numerical Path feature, which can be interpreted as a *string*.

General idea

- Put in the *same cluster* individuals $\mathbf{x}_1$ and $\mathbf{x}_2$ which are *similar*
- Put in *different clusters* individuals $\mathbf{x}_1$ and $\mathbf{x}_2$ which are *dissimilar*

Definition of a *dissimilarity* $\delta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$

- Separation: $\forall (\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{X}^2, \delta (\mathbf{x}_1, \mathbf{x}_2) = 0 \Leftrightarrow \mathbf{x}_1 = \mathbf{x}_2$
- Symmetry: $\forall (\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{X}^2, \delta (\mathbf{x}_1, \mathbf{x}_2) = \delta (\mathbf{x}_2, \mathbf{x}_1)$

Definition of a *distance* $\delta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$: it is a dissimilarity with Triangular inequality: $\forall (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) \in \mathcal{X}^3, \delta (\mathbf{x}_1, \mathbf{x}_3) \leq \delta (\mathbf{x}_1, \mathbf{x}_2) + \delta (\mathbf{x}_2, \mathbf{x}_3)$

Definition of a *similarity* $s : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$

- $\forall \mathbf{x}_1 \in \mathcal{X}, s(\mathbf{x}_1, \mathbf{x}_1) = s_{\max}$ with $s_{\max} \geq s(\mathbf{x}_1, \mathbf{x}_2)\ \forall (\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{X}^2, \mathbf{x}_1 \neq \mathbf{x}_2$
- Symmetry: $\forall (\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{X}^2, s(\mathbf{x}_1, \mathbf{x}_2) = s(\mathbf{x}_2, \mathbf{x}_1)$

3.1.4 Clustering Approaches

In this section, we present and discuss the application of unsupervised learning techniques to analyze categorical data in a large-scale dataset . A crucial aspect to be considered in unsupervised learning is how to partition the data in subsets containing similar elements within them. At the same time, these subsets have to contain ensembles of elements as dissimilar as possible from each other. This particular aspect is strictly related to the possible optimal selection of the k value that represents the number of clusters in which the dataset is partitioned. Generally, the k value is the input for a clustering algorithm. As it is well known, the k selection is still an open research problem, and the notion of optimal should be re-stated as more suitable or recommended with respect to the research domain, the available data and the main goals of the Data Analysis. In this section we present a strategy for the k selection and discuss the clustering algorithms within an experimental scenario.

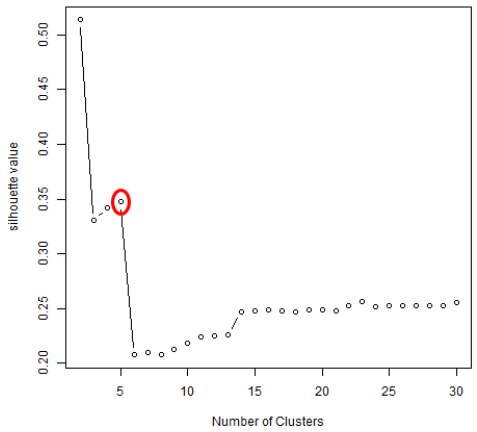
For what concern the unsupervised classification, we focus on the following two methodologies: i) the k-medoids algorithm; ii) the hierarchical clustering.
The k-medoids is a partitioning technique that clusters the dataset of n objects into k clusters. In details, *a medoid* is a selected point from the dataset, whose average dissimilarity with respect to all the data points in a cluster is minimal; in other words it is the most centrally located point in the cluster. In our case, the medoid represents a specific visitor's path in the museum and it can provide a synthetic description of users having a similar behaviors. To perform the experimental scenario we use some R packages: `kmed`[19], `hclust` and `cutree`[109].

Another suitable classification approach for dataset containing categorical data is the Hierarchical Clustering (HC). HC is a hierarchical decomposition of data based on group similarities. Here we consider the Ward's linkage method (WM). WM expresses the distance between two clusters, C_1 and C_2, by getting how much the sum of squares will increase when clusters are merged. By this, Ward is less susceptible to noise and outliers. Apart form former implementation issues in the R framework, the used approach is explained in[88], and has been used Ward2 (`ward.D2`), which minimizes:

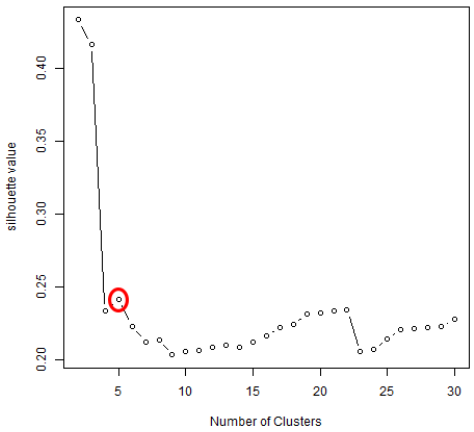
$$D(C_1, C_2) = \frac{|C_1|\,|C_2|}{|C_1| + |C_2|}\,\|C_1 - C_2\|^2 \tag{3.5}$$

3.1.5 The k selection strategy

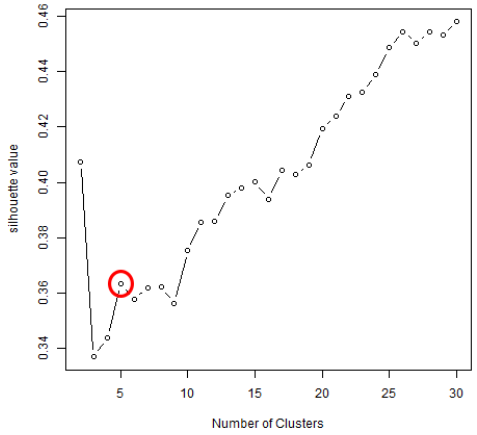
In order to choose the k value as input for the clustering algorithms, we used the `NbClust` package (see[25]). As stated in[68] quality indexes are mostly based on the concepts of dispersion of a cluster and dissimilarity between clusters. The most popular indexes are the Dunn index, the Davis-Bouldin index, and the Silhouette index. Among all these indexes, the Silhouette one gives the best results accordingly to previous research[33;34;105] in the Cultural Heritage domain. Figure 3.5 reports the silhouette's results considering the four similarity and distance functions previously described. High silhouette values indicates that clustering solution is feasible. For each selected index, the related k value witch ranges from 2 (the minimum number of clusters) to 30 (a rough estimate of the maximum number of possible clusters) has been explored. Then in order to obtain the k value, the local maximum for Silhouette has been evaluated. The aim is to identify the most appropriate number of clusters for the data, so *difference-like criteria*, and similar *optimization-like criteria* have been applied, according to[140]. In particular instead of looking only at two consecutive data partitions (formed by k and $k+1$ clusters), we looked at the triple $(k-1, k, k+1)$. In order to select the most appropriate k, with reference



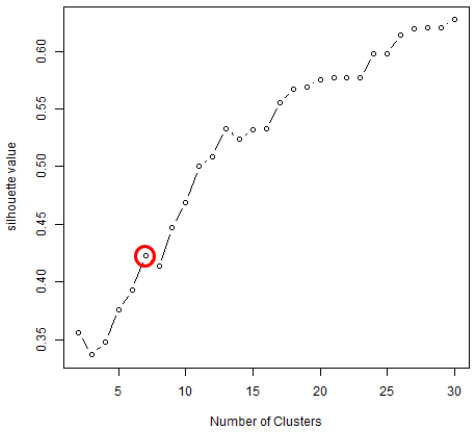
(a) silhouette index for Lv distance



(b) silhouette index for LCS distance



(c) silhouette index for cosine distance



(d) silhouette index for Jaccard distance

Figure 3.5: Results from NbCluster using silhouette indices with a Ward D2 linkage (WM) and the four distances. (a), (b) and (c) suggest a $k = 5$ while (d), which describe the result for the Jaccard distance, suggest $k = 7$ with an higher silhouette value.

to Figure 3.5 we selected the first local maximum. In this case, three distances suggest $k = 5$ (Fig. 3.5). Moreover, this choiche results to be in accordance with our past research focused on visitor behaviours studies in the CH domain[105].

3.1.6 Supervised Learning

There is a long list of possible MLA, in this section we will attempt to show some of them by grouping them according to their functional similarity (how they work).

Based on the work of[18]. There are still algorithms that could be classified as belonging to more than one category, there are also similar-sounding categories that describe the problem and the class of algorithm, such as Regression and Clustering, the two most common supervised machine learning problems.

Algorithms Based on Instances A decision problem with instances or examples of training data that are deemed important or required by the model is an instance-based learning model. Such methods typically create a database of example data and use a similarity measure to compare new data to the database in order to find the best match and make a prediction. As a result, instance-based methods are also referred to as winner-take-all methods and memory-based learning. The most widely used instance-based algorithms: k-Nearest Neighbor (kNN), SOM (Self-Organizing Map), LWL (Locally Weighted Learning), SVM (Support Vector Machines),

Algorithms with Bayesian Inferenece Bayesian methods are those that use Bayes' Theorem explicitly to solve problems like classification and regression. The most widely used Bayesian algorithms are: Naive Bayes, Gaussian Naive Bayes, Multinomial Naive Bayes, Bayesian Network (BBN)

Algorithms for Clustering Clustering, like regression, describes the problem class and the method class. Modeling approaches such as centroid-based and hierarchical clustering are commonly used to organize clustering methods. All methods are concerned with using the inherent structures in the data to best organize the data into groups with the greatest possible commonality. The most widely used clustering algorithms are: Expectation Maximisation (EM), Hierarchical Clustering, k-Means, k-Medians

Algorithms based on Artificial Neural Networks Artificial Neural Networks (ANNs) are models inspired by the structure and/or function of biological

neural networks. They are a type of pattern matching that is commonly used for regression and classification problems, but they are actually a vast subfield with hundreds of algorithms and variations for a wide range of problem types. The following are the most widely used artificial neural network algorithms: Multi-layer Perceptron (MLP), Radial Basis Function Network, Stochastic Gradient Descent.

Algorithms for Dimensionality Reduction Dimensionality reduction methods, like clustering methods, seek and exploit the inherent structure in data, but in this case in an unsupervised manner or in order to summarize or describe data with less information. This can be useful for visualizing dimensional data or simplifying data for use in a supervised learning method. Many of these techniques can be adapted for classification and regression: Principal Components Analisis (PCA), Linear Discriminant Analysis (LDA), Quadratic discriminant analysis (QDA).

Ensemble Algorithms Ensemble methods are models that are made up of multiple weaker models that have been trained independently and whose predictions are then combined in some way to make the overall prediction. Much thought is given to which types of weak learners to combine and how to combine them. This is a very powerful technique class, and as such, it is very popular: Increasing Bootstrapped Aggregation (Bagging), Weighted AdaBoost Average (Blending), Generalization Stacking (Stacking), Machines for Gradient Boosting (GBM), Regression Trees with Gradient Boosting (GBRT), Random Forest. X-Gradient Boosting (XGBoost).

Let us look into detail at the three models: MLP, RF, and XGB. The Multi-layer Perceptron (MLP) is a feedforward neural network able to solve nonlinear problems, composed of multiple perceptrons disposed in multiple layers. The random forest, introduced by Breiman, is essentially a bagged decision where sub-sampling is applied to decision trees at each split, by averaging the results

of the tree. Bagging is a typical ensemble technique known also as bootstrap aggregating. Even if it was designed to be applied to classification is used also for Regression problems.

3.2 Knowledge Graph Embedding

3.2.1 KG database

This paragraph depicts and organizes some definitions related to KG terms used in this paper. KG definitions have been addressed in many research works, such as[40;53;98], but there is no common agreement on such terminology. We consider a KG to be *a technology used to acquire and integrate information into an ontology and to apply a reasoner to derive new knowledge*[40], this *ontology creating a set of interconnected typed entities and their attributes*[48]. From this point of view, we have considered and implemented an Enterprise KG (EKG), also reviewed in[48;53], that considers KG in companies, within a specific business domain. Moreover, KG Management Systems (KGMS) are recognized to be systems developed to exploit KG[10;11]. This paper aims to propose a process that results in an EKG Management System and to describe its application to a real-world problem.

The reasons why organizations choose graph databases consist of their performance, responsiveness to queries, agility and flexibility for real-world business problems, as stated in[86;112]. When we have data organized in rows and columns, relations can be expressed between different table joins and column groupings. The connections between cells are important concerning future business needs, and the data in such a table structure raise a connected data problem, namely how to make them more informative. Whereas in a graph database the relationship takes the priority[86]. In a relational database, essentially, each row is an entity. A relational database has a ledger-structure. When we use foreign keys, we aim

to create relations among those attributes. Relational databases are also useful for Big Data and fast queries. However, queries can be complex when multi-level joins are required. It is very far from how the real world is considered and how this is mapped to a database. Accordingly, the main problem is how to re-create schemas for specific business questions. In fact, unlike the relational databases, where join-intensive query performance deteriorates as the dataset increases in size, with a graph database the performance tends to remain relatively constant, even as the dataset grows[112].

A comprehensive study on the most common graph databases has been performed in[96], the majority of the presented comparisons are focused on more general use of graphs. In the mentioned survey, Neo4j proved a good choice for the storage, processing and analysis of such datasets. A solution that focuses on KG is Grakn. Grakn Labs defines its product Grakn as a distributed *hyper-relational* database for knowledge-oriented systems, since it implements *entity-relation* paradigms to manage complex data structures and ontologies. Grakn builds on top of the columns store NoSQL Cassandra database and extends it with a software layer that helps to build and manage a KG quickly. The peculiarity of Grakn is that it is a graph database that builds its characteristics on top of the column-family database model Cassandra.

Recent literature is being more and more interested in this novel solution. In[13] the authors propose a framework with Grakn, it is used in biomedical applications[21;113]. In[125] the authors recognize in Grakn a suitable data model for a hypergraph database that addresses the problem of aligning graph and relational databases and coherently handles hyper-elements, i.e. generalized relationships, such as the one proposed in their work. Furthermore, in[65] Grakn is recognized as a *newcomer* without any significant benchmark in the academic literature, and in this recent survey on KG,[55], as a tool for integrating knowledge graph with machine learning technique.

3.2.2 Knowledge Graphs

Assembling and managing complex, highly interconnected data as knowledge graphs poses unique knowledge and data representation challenges[40].

Other than those carried over from the XML datatype definitions, RDF does not support any semantics on its own. SPARQL is a querying language for RDF graphs that is natively implemented by triple stores.

Applications built on knowledge graphs must work efficiently with semantically rich but well-structured graph data. While relational modelling and graph databases can help with some specific issues, they cannot provide a complete technical and conceptual framework. Many instead turn to Semantic Web standards, particularly the popular Web Ontology Language (OWL), as a "silver bullet" for managing semantic graphs. While the Semantic Web stack is extremely useful for publishing linked data on the web, its value as a knowledge graph representation solution for standalone domain-specific applications is less clear.

W3C initiative started in late 1990s to extend existing web architecture with a layer of formal semantics. This layer enables machines to intelligently share and interpret data globally. The W3C's technology stack includes three data and knowledge representation standards: RDF, RDFS, and OWL.

RDF

RDF is a graph-based data model. It represents data as a directed multigraph with labelled vertices and edges (multiple edges with different labels between the same nodes are allowed). For convenience, IRIs (abstract "things") are represented as well as literals (concrete data values). Graphs are expressed as a set of triples, each interpreted as a "predicate" edge from the "subject" to the "object". RDF is a data model and does not support any semantics beyond those defined in the XML datatype definitions. In order to query RDF graphs, triple stores, i.e. databases designed to store and manage RDF data, natively implement SPARQL. On top of a

triple store, the Wikidata project exposes RDF data via a live SPARQL endpoint.

A class and property subtype, as well as property range and domain restrictions, are all part of the RDFS (RDF Schema). Simple type hierarchies can be built over RDF data, and are represented within RDF graphs. Since SPARQL can capture the effective reasoning mechanism over RDFS (via property paths), additional computationally expensive inference tools are not required.

OWL

OWL (Web Ontology Language) is a family of description logic-based ontology languages. OWL extends RDFS' ontological constructs. Aside from reasoners (OWL DL, EL), rule engines (OWL RL), and query rewriting systems (OWL QRS), OWL ontologies can be represented in RDF graphs (OWL QL).

Unlike relational database systems, OWL uses the open-world assumption (OWA), which means that a lack of information is not interpreted as if it is false. According to the OWL constraint "Every parent must have at least one child", the dataset contains the single fact "John is a parent", but not John's children. Unless otherwise stated, we can safely assume John has a child, even if we are unaware of it. This ethos fits well with the open-ended web environment, where incomplete information is assumed.

While the use of RDF(S) standards for web data publishing has increased in recent years, OWL usage has been surprising limited [2], [3]. This is true for both the number of applications and specific ontological constructs that have been successfully used. Ordnance Survey, the UK's national mapping agency, uses expressive OWL ontologies to structure geographical and administrative data. Some of the reasons for this phenomenon have encouraged our company to keep looking for a better knowledge representation solution, as explained in the next section.

GRAKN

The GRAKN.AI knowledge graph platform seeks to address a gap in current knowledge graph representation paradigms. Grakn is built on Apache TinkerPop, an open-source interface that provides consistent access to data stored in any TinkerPop-enabled database. Grakn's underlying data structure is a labelled hypergraph. This is then mapped to a labelled, directed graph, which TinkerPop exposes regardless of the actual data storage involved.

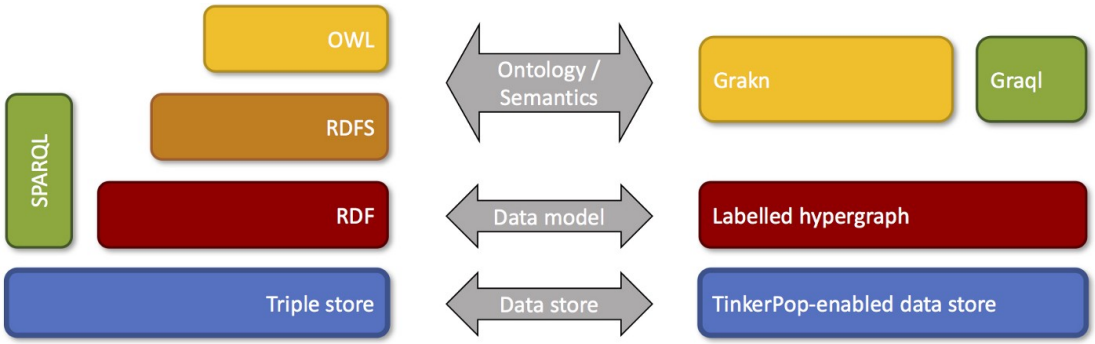


Figure 3.6: Labelled, directed multigraphs are the foundation of the RDF data model, so mapping RDF to hypergraphs is straightforward. Grakn exposes a higher level knowledge model than OWL, allowing developers to represent their application domain in terms of entities, resources, relations, and roles, rather than individuals, literals, properties, and classes. Image Credits:[64].

Grakn is storage-agnostic, working with graph databases and triple stores such as Titan, OrientDB, Blazegraph, StarDog, and others. In contrast to OWL's individuals, literals, properties, and classes, Grakn allows developers to represent their application domain in terms of entities, resources, relations, and roles. We carefully combine both styles of reasoning in Grakn, taking the best of both worlds: open-world inference and closed-world constraint checking. In theory, the OWL architecture allows for the use of arbitrary fragments (as needed on per use-case basis). In practice, the nature of the available reasoning tools makes "cherry picking" difficult.

According to[64], this open-source interface allows for uniform access to data

stored in any TinkerPop-enabled database.

This architecture has two immediate benefits:

- Grakn is a labelled hypergraph that can work on top of graph databases and triple stores like Titan, OrientDB, Blazegraph, StarDog, and others that implement the TinkerPop interface.
- This is then mapped to a labelled, directed graph, which TinkerPop exposes regardless of the data storage involved.

A comparison of architectures ins presented in

Here are four reasons why[64] thinks that Grakn ontologies are better suited to modelling knowledge graphs than OWL:

- Open and Closed World Assumptions in Grakn
- Inconsistent expressiveness/complexity ratio in OWL profiles
- OWL not suitable for complex graph reasoning and higher difficulty

Open and Closed World Assumptions Grakn By using the OWA, OWL makes it difficult to validate data consistency and structure. That's what knowledge graph applications need, just like relational databases need strict schemas to ensure data quality.

Grakn combines the best of both worlds: ontological-style open-world inference and schema-like closed-world constraint checking. Antagonism between open-world "ontological" and closed-world "schema" modelling stems from more than just formal differences. Contrast an open-ended, heterogeneous web of data with tightly curated single-view data stores. Because we work with large domain-specific knowledge graphs, we find that both ends of this spectrum are too restrictive and need to be balanced.

Inconsistent expressiveness/complexity ratio in OWL profiles None of the standard OWL profiles directly match typical knowledge graph schema/ontology requirements. These rich constraint patterns are only available to a limited extent in OWL DL, i.e., the most complex of the decidable OWL profiles. However, the expressiveness of the lightweight OWL QL, OWL RL, or even RDFS is sufficient in this regard.

The OWL architecture allows for arbitrary fragments (as needed on per use-case basis). In practice, however, "cherry picking" is limited by the nature of available reasoning tools, which must account for entire OWL profiles. The two simple constraints "Every parent has a child" and "Every child is a person" require a full-fledged OWL DL reasoner, which scales poorly with large datasets. This frequently forces Semantic Web practitioners to rely solely on RDF(S), which is too simplistic as an ontology/schema language on its own.

OWL not suitable for complex graph reasoning and higher difficulty For managing tree-shaped data, its formal foundations (logics with the so-called tree-model property) are determined largely by computational constraints (predominantly decidability). As a result, the complexity/expressiveness overhead required to work with OWL does not return value in knowledge graphs.

Because OWL is based on description logic from research, non-logicians must be able to comprehend the language and achieve the intended behavior of OWL-backed systems. This is another reason many developers prefer RDF (S).

Grakn hope to reach a much larger audience than OWL by keeping Grakn's knowledge representation formalism lightweight and built bottom-up from developer feedback.

Grakn had to be equipped with a new, dedicated query language, Graql, in order to provide optimal access to information represented in Grakn knowledge graphs by committing to a novel ontology formalism.

The creation of a practical yet well-founded knowledge representation formalism

is a difficult task that necessitates careful consideration of a wide range of issues involving formal, knowledge engineering, and technological perspectives.

3.2.3 Graph embedding

Machine learning techniques take as input a feature vector where the learning task is to map this feature vector into a prediction output that includes a latent vector (embedding) or class labels, regression score, or an unsupervised clustering[92]. Applied to KG, the input of these machine learning methods are the graphs, and this section will focus mainly on the embedding output. One of the first knowledge graph embedding methods is RESCAL[93] which uses a tensor factorization method to compute an adjacency tensor that represents the knowledge graph. Embedding is the best-known technique to pass from a graph and express its nodes and links in a low-dimensional vector space, usually $\mathbb{R}^K$. An extensive literature review on embedding techniques can be found in[20;91;92;123]. With the embedding method, *similar* nodes are mapped in close points in the space. The way in which this process is performed defines the specific algorithm.

Well known embedding algorithms to be mentioned include TransE and TransR[54], i.e. *transaction-based approaches*. These kinds of models are characterized by measuring the "plausibility of a triple" as it can be understood as a distance between the entities. In this case, each entity is modelled as a point in a vector space and each relation as a translation and projection. TransE is similar to word2vec, in the sense that giving a triple (subject, relation, object) when the subject is similar to the object entity, then the respective embedding should be close to the object embedding. The authors in[1;72] propose CETransR which extends TransR first by modelling the entities and relations in distinct vector spaces and then by clustering the same knowledge mentions into groups. A second family is *semantic matching model* this kind of models use a score function-based. The first example is the RESCAL model. A model directly linked to RESCAL is

DistMult, that is RESCAL with a diagonal matrix, that is particularly suitable for symmetric relation. By combining some proprieties of RESCAL and DistMult, HolE can be obtained. It takes relation and entities as vector and uses the circular correlation operation to compose the entity representation. An improvement over DistMult is ComplEx. ComplEx[50;137] is based on a *complex* embedding that can handle a *large* variety of binary relations. It is *scalable* to large datasets as it remains linear in both space and time. Furthermore, it is *simpler* concerning other alternatives, as it only uses the Hermitian dot product. Given a fact $r(s, o)$, a relation r between a subject s and an object o, their respective embedding e_s and e_o belong to $\mathbb{C}^K$, where K is the desired embedding space dimension. A different extension of RESCAL is ConvE[37], where the idea is to shape with convolutional layers entities and relations. Most recent graph embedding solutions are TransC[77], RotatE[132] (that is a kind of translational model), QuatE[156], NSCaching[157], DeepPath[147], and CoKE[142].

To use most of the embedding in a single project, nowadays there are *graph embedding systems* for using multiple embedding: AmpliGraph[31], GraphVite[158], and Pykg2vec[152].

Network embedding visualization The purpose is to map each graph element in a low-dimensional space. First, we produce dispersed representations for the nodes that we can re-use in other tasks, and then, for example, if we had this vector representation, we can go and evaluate the similarity between those embeddings. Typically, if these vector representations are comparable, we want the nodes to be similar as well. As a result, they could be neighboring in a graph, related, or have comparable structural roles in the network. (Fig. 3.7) Ultimately, we want each node representation to encode as much network information as feasible.

We wish to transition from a graph representation to a vector-space representation, hence we need a mapping function that goes from the node to the embedding.

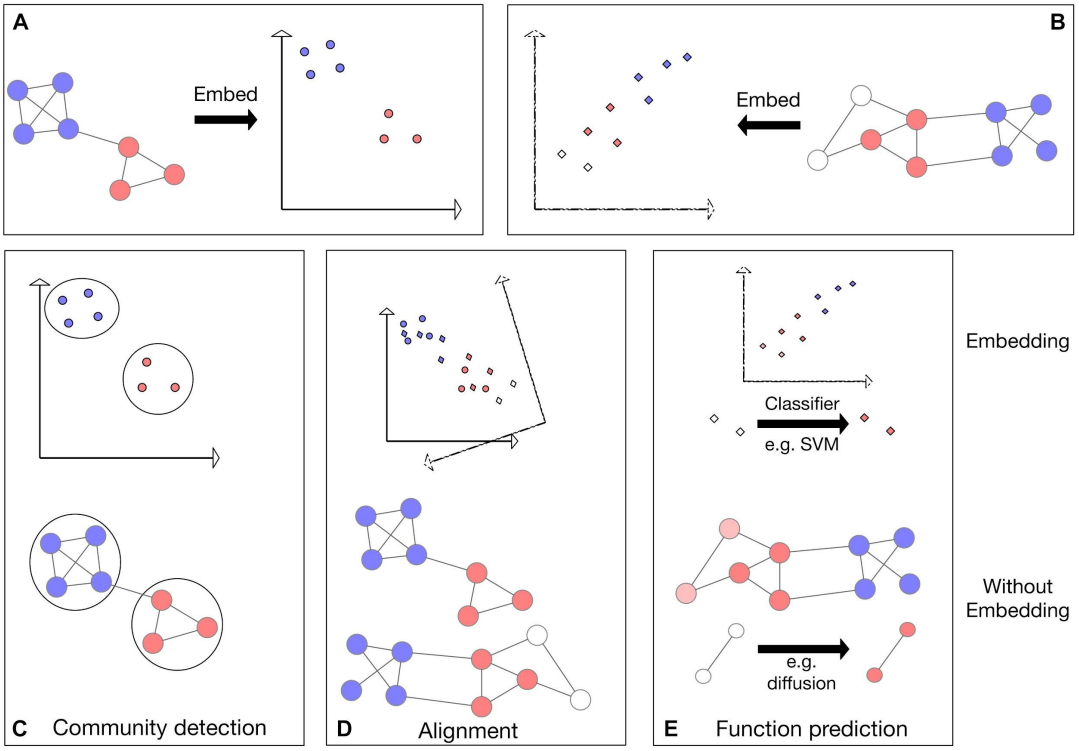


Figure 3.7: Three applications for network embeddings shown schematically. Colors represent network node features. (A,B) 2D embedding of two networks. (C) A community detection visualization in embedded space and on the network (bottom).
(D) Network alignment visualization in embedded space. As shown in panel (B), the network embedding is rotated, translated, and reflected to align with the embedding in panel (A). Bottom of (D): direct alignment of two networks visualized as vertical proximity. (E) Embedded function prediction visualization Nodes (bottom) and embeddings (top) previously unlabeled (white) and then labeled (colored). Credits: [90]

This embedding would also be an ideal dimensions space. One issue is deciding on the number of dimensions, because depending on the size of the space, you will be able to fit more or less information there. Because we don't know what these attributes represent, the axes aren't labeled. This is the type of mapping we wish to do.

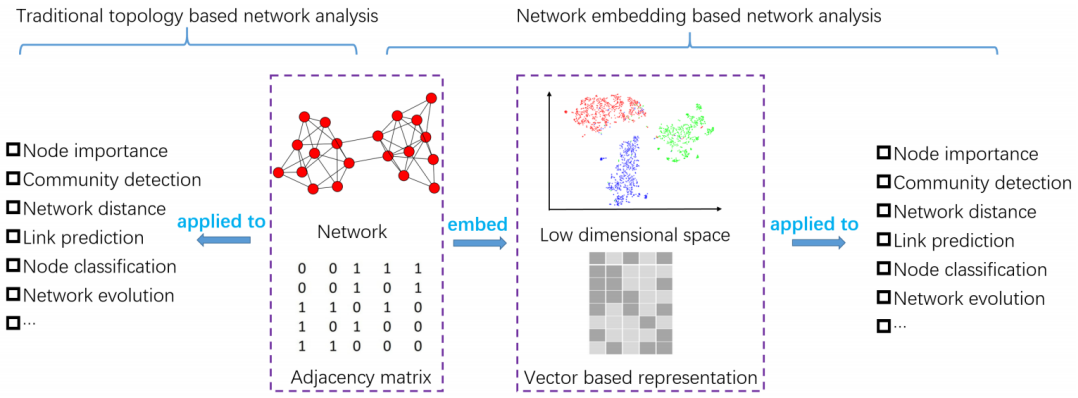


Figure 3.8: Network topology vs Network embedding Credits: [20].

Network topology vs Network embedding We begin with a huge graph and then construct the adjacency matrix. And this is the type of representation that we dislike working with because it is both sparse and huge. As a result, maintaining it as a structure is prohibitively expensive. Instead, we desire a more compact representation in which the columns no longer reflect all of the possible nodes in the network. They are, however, what we refer to as latent dimensions. As a result, each of these dimensions will be able to capture part of the characteristics of that node. The important issue is that these dimensions are latent, not explicit. The tasks that you can work on after you have this distributed representation are the same as those that you can do with the graph representation. It can be used for anomaly detection, feature prediction, node attribute prediction, clustering, link prediction, and so on. As a result, these embedding approaches are exceedingly powerful.

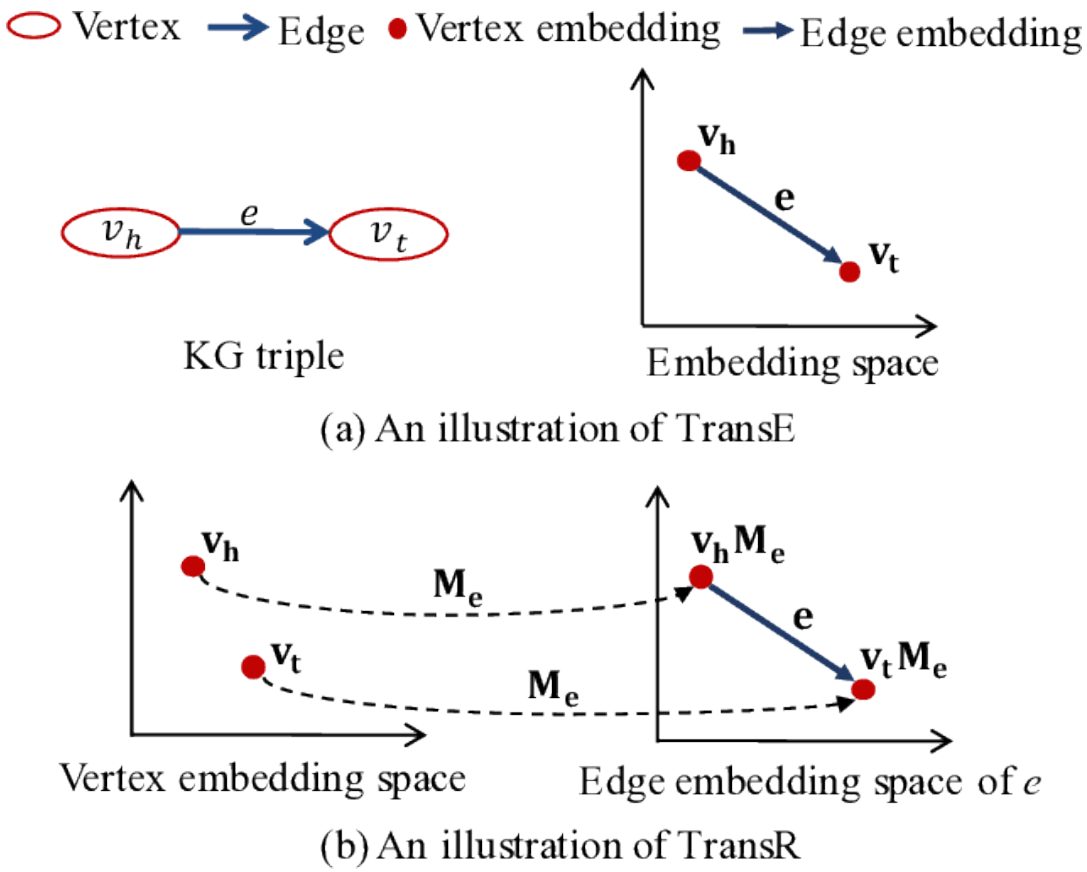


Figure 3.9: Translational embedding: TransE, TransR Credits: [143]

TransE There are numerous Knowledge Graph Embedding techniques in the literature, and picking the best one for a given dataset is a significant difficulty. The TransE embedding is one of the most basic. It is predicated on the premise that one entity in a triple is shifted by its relationship to another entity. As a result, this method optimizes the alignment of all entities depending on their relationships. And so, it is a Translational embedding algorithm. And the score function that was utilized to construct the embedding is : $||(e_s + w_r) - e_o||_p$.

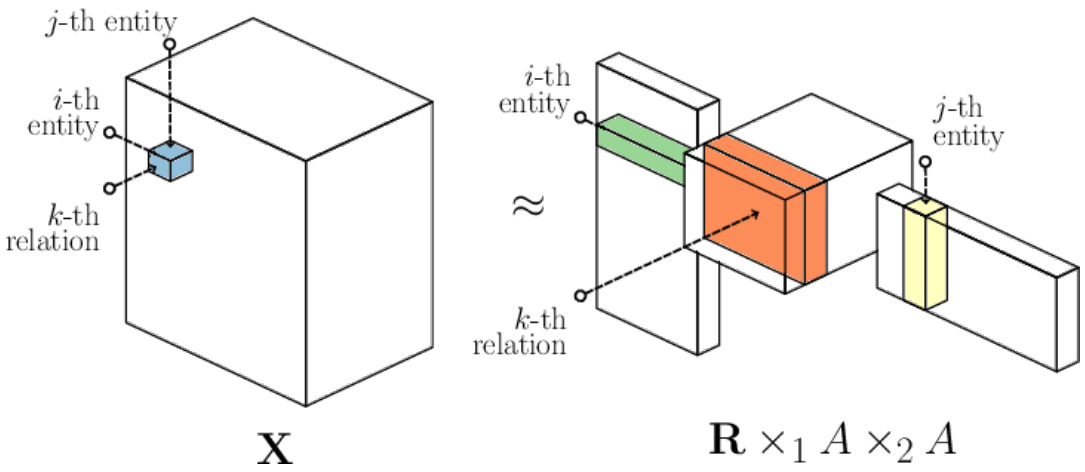


Figure 3.10: RESCAL binary relations model Credits:[66]

ComplEx Bilinear embedding is another type of embedding algorithm. It is based on the tensor space decomposition. CompleEx, DistMult, and RESCAL are a few examples. The ComplEx embedding's score function is: $\Re(< w_r, e_s, \bar{e}_o >)$.

It is not always easy to obtain a high-quality model when there is enough data to generate valuable and appropriate insights. Each Knowledge Graph Embedding algorithm has unique input parameters that must be configured. In this case, we used a random grid search to discover the best option for our data.

To summarize, unsupervised learning algorithms, such as clustering, used on the Knowledge Graph Embedding enable the discovery of hidden knowledge that would have been more difficult to discover using the original legacy dataset structure. (Fig. 3.7)

Chapter 4

Data engineering methodologies

This chapter addresses the five W's: who, what, why, when, and how, among other things. It explains what I was able to accomplish in terms of the research objectives. Reasons for selecting this particular strategy above others. What tools I used to collect data and why I utilized them, as well as when the data was obtained and in what format it was collected. What tools I used to evaluate the data and why I employed them, and lastly, what ethical considerations should be taken into mind when dealing with such sensitive data.

In order to obtain the results in the following sections, we utilized a pipeline, a data science garage technique, which is a mixture of IBM data since, and in my opinion best handles any sort of data since application process, while also documenting. The project relies on IoT sensors, data from SQL databases. We must explore the tools in order to collect data for analysis. Or other business intelligence tools like as Power BI and Tableau. Whereas we also explore various algorithms for image recognition, such as SWIFT.

4.1 Data Collection

In general, a non-invasive Bluetooth tracking system works as follows: when a Bluetooth-activated mobile device enters the detectable area (the surrounding area of a Bluetooth board), a sensor receives the emitted signal from the mobile device until the signal disappears.

4.1.1 Museums environment

Collecting data about visitors' movements within a cultural space is especially useful for:

- analyzing and understanding visit routes;
- analyzing and understanding length of stay;
- identifying elements (artworks, shop windows, panels, captions, etc.) that attract or are ignored.

Tracking and influencing museum visitors' behavior is critical for building an engagement relationship and improving the user experience.

In this context, it is critical to know not only the number of users who purchased a ticket, but also qualitative data about the experience. How long were they in the museum? What is the rate of return? How many works have they seen in total? How much did they learn during their visit?

In an attempt to answer some of the preceding questions, we present and discuss a Machine Learning methodology for classifying IoT cultural data using unsupervised techniques in this paper. Such information was gathered in a real-world experimental setting, the National Archaeological Museum of Naples. We collected non-invasive behavioral data about visitors using an IoT framework comprised of Bluetooth sensor boards.

The collected dataset, which was mostly made up of categorical data, was clustered using two approaches appropriate for this type of data.

MANN

In order to assess the proposed approach we collected the behavioural data by monitoring the National Archaeological Museum of Napes, where many rooms are distributed into four floors (see Fig. 4.1). The museum environment has been equipped with IoT sensor boards, with Bluetooth capabilities; each board has been placed in a key place, covering all the available location. Sixteen Bluetooth sensor boards have been deployed within the museum environment acting as a non-invasive Bluetooth tracking system. Figure 4.1 presents the locations of the sensor boards represented by blue circles. These locations have been mapped with useful labels (nodes E, H, C, U, F, B, C, G, Y, M, V, P, S, T). In details, the framework is composed of smart sensor boards with Bluetooth and Wi-Fi capabilities; each board has been located inside a museum room covering the majority of the available locations. It acts as a non-invasive proximity tracking system: when a Bluetooth-activated mobile device enters the detectable area (the surrounding area of a sensor board), its emitted signal is collected. Therefore, the IoT system is able to track a visitor path by collecting his position and the related time-of-stay inside the museum rooms. Fig. 1 presents the position of the sensor boards, labelled as follows: . All the IoT boards collect, store and sent to a cloud server the data about users' presence inside the museum. Then data are joined giving as output the visiting paths as temporal sequences of labels (e.g. EHCFYMVPCHE is a path of a visitor starting from the E location and finishing in the same location).

The Data collection task has been performed from August to December 2018. Then a data clean-up and processing step has been executed in order to remove any inconsistencies. Our dataset relies on over than 9700 unique users behavioural data

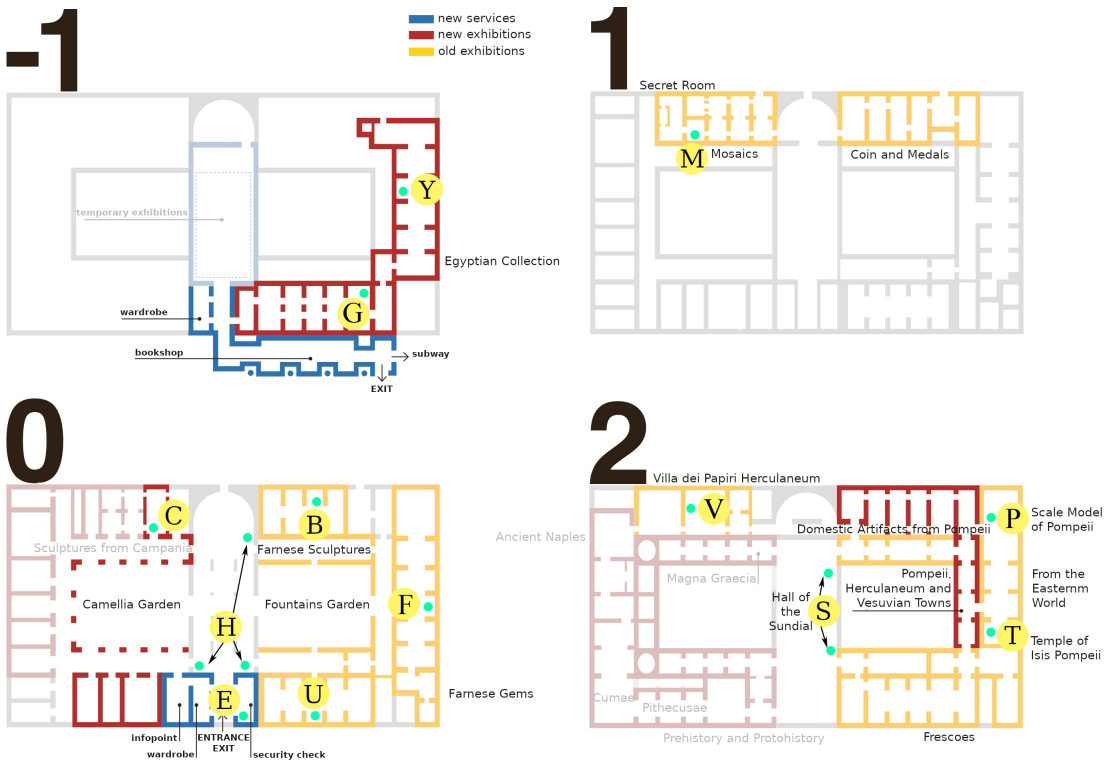
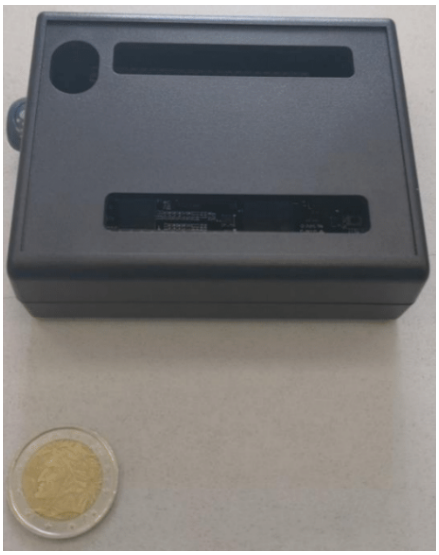


Figure 4.1: A 2D map of the museum rooms and floors where the IoT system has been deployed. The Museum map composed by four floors. The pale blue circles represent the sensor boards location with a label. The closed and not available rooms of the museum have been also reported.

composed by two features: (i) the visiting *Path* (a non-numerical feature) and (ii) the length of stay (a numerical feature). In this case, a string is a concatenation of labels, where each label represents (see Figure 4.1) a sensor node located within the environment.

Collect Raw Data: IoT sensors One of the sensors is visible in this image. It is made up of smart sensor boards that have Bluetooth and Wi-Fi capabilities. The goal of the second stage of data science is to locate all available data and collect it in usable formats (as .csv, .json. .xml) We can see an xls file with all of the data here. This is a log file, and each row represents the time stamp at which a sensor "sees" a specific visitor via an id. As a result, the IoT system may follow a visitor's trip by gathering his position and time-of-stay inside the museum rooms.



TIMESTAMP	CentralinaCorrente	OraCentralinaPrecedente	CentralinaPrecedente	deltaTime	ID_VISITATORE
09/09/2018 10:51:56	Ingresso			0	id_0000001
09/09/2018 10:52:09	Atrio alto destro	09/09/2018 10:51:56	Ingresso	13	id_0000001
09/09/2018 10:52:21	Ingresso	09/09/2018 10:52:09	Atrio alto destro	12	id_0000001
09/09/2018 10:52:34	Atrio alto destro	09/09/2018 10:52:21	Ingresso	13	id_0000001
09/09/2018 10:52:47	Atrio alto destro	09/09/2018 10:52:34	Atrio alto destro	13	id_0000001
09/09/2018 10:52:59	Atrio alto destro	09/09/2018 10:52:47	Atrio alto destro	12	id_0000001
09/09/2018 10:53:12	Atrio alto destro	09/09/2018 10:52:59	Atrio alto destro	13	id_0000001
09/09/2018 10:53:24	Sala la Farnesina	09/09/2018 10:53:12	Atrio alto destro	12	id_0000001
09/09/2018 10:53:37	Sala la Farnesina	09/09/2018 10:53:24	Sala la Farnesina	13	id_0000001
09/09/2018 10:53:50	Sala Farnese	09/09/2018 10:53:37	Sala la Farnesina	13	id_0000001
09/09/2018 10:54:02	Sala Farnese	09/09/2018 10:53:50	Sala Farnese	12	id_0000001
09/09/2018 10:54:15	Ingresso	09/09/2018 10:54:02	Sala Farnese	13	id_0000001
09/09/2018 10:54:28	Ingresso	09/09/2018 10:54:15	Ingresso	13	id_0000001
09/09/2018 10:54:41	Atrio basso sinistro	09/09/2018 10:54:28	Ingresso	13	id_0000001
09/09/2018 10:54:53	Ingresso	09/09/2018 10:54:41	Atrio basso sinistro	12	id_0000001
09/09/2018 10:55:06	Ingresso	09/09/2018 10:54:53	Ingresso	13	id_0000001
09/09/2018 10:55:19	Ingresso	09/09/2018 10:55:06	Ingresso	13	id_0000001
09/09/2018 10:58:55	Ingresso	09/09/2018 10:55:19	Ingresso	216	id_0000001
09/09/2018 10:59:08	Atrio basso sinistro	09/09/2018 10:58:55	Ingresso	13	id_0000001
09/09/2018 10:59:20	Atrio alto destro	09/09/2018 10:59:08	Atrio basso sinistro	12	id_0000001
09/09/2018 10:59:33	Ingresso	09/09/2018 10:59:20	Atrio alto destro	13	id_0000001
09/09/2018 10:59:34	Atrio alto destro	09/09/2018 10:59:33	Ingresso	1	id_0000001
09/09/2018 10:59:59	Atrio alto destro	09/09/2018 10:59:34	Atrio alto destro	25	id_0000001
09/09/2018 11:00:12	Sala Mosaici	09/09/2018 10:59:59	Atrio alto destro	13	id_0000001
09/09/2018 11:00:25	Sala Mosaici	09/09/2018 11:00:12	Sala Mosaici	13	id_0000001
	Atrio alto destro	09/09/2018 11:00:25	Sala Mosaici	0	id_0000001
09/09/2018 11:00:38	Sala Mosaici	09/09/2018 11:00:25	Atrio alto destro	13	id_0000001
09/09/2018 11:00:50	Sala Mosaici	09/09/2018 11:00:38	Sala Mosaici	12	id_0000001

Figure 4.2: Photo of the sensor and sample of the extracted log data.

Process the Data This log data must be cleansed in order to yield useful insights.

The goal at this stage is to understand the data, seek for errors, missing values, corrupted records, and, if possible, clean the data. For example, in our data, visors

may have turned off their phone Bluetooth at some point during their visits, or there may be a large number of people near the sensor, and it may not be able to retrieve/receive them all at once, and some sensors may not be working or may be switched due to power issues. The Bluetooth signal collects data from personnel inside the museum who are not actually following a visitor's walk.

As a result of this cleaning, we obtain a path, i.e. a sequence of letters indicating where the phone has been, as well as the total time spent in the museum, as well as the time spent at each of the nodes. We have a lot of information, i.e. n=20,000 visitors, and 16 features, 1 of which is a string type (path), and 15 of which are time values.

TIMESTAMP	CentralinaCorrente	OraCentralinaPrecedente	CentralinaPrecedente	deltaTime	ID_VISITATORE
09/09/2018 10:51:56	Ingresso			0	id_0000001
09/09/2018 10:52:09	Atrio alto destro	09/09/2018 10:51:56	Ingresso	13	id_0000001
09/09/2018 10:52:21	Ingresso	09/09/2018 10:52:09	Atrio alto destro	12	id_0000001
09/09/2018 10:52:34	Atrio alto destro	09/09/2018 10:52:21	Ingresso	13	id_0000001
09/09/2018 10:52:47	Atrio alto destro	09/09/2018 10:52:34	Atrio alto destro	13	id_0000001
09/09/2018 10:52:59	Atrio alto destro	09/09/2018 10:52:47	Atrio alto destro	12	id_0000001
09/09/2018 10:53:12	Atrio alto destro	09/09/2018 10:52:59	Atrio alto destro	13	id_0000001
09/09/2018 10:53:24	Sala la Farnesina	09/09/2018 10:53:12	Atrio alto destro	12	id_0000001
09/09/2018 10:53:37	Sala la Farnesina	09/09/2018 10:53:24	Sala la Farnesina	13	id_0000001
09/09/2018 10:53:50	Sala Farnese	09/09/2018 10:53:37	Sala la Farnesina	13	id_0000001
09/09/2018 10:54:02	Sala Farnese	09/09/2018 10:53:50	Sala Farnese	12	id_0000001
09/09/2018 10:54:15	Ingresso	09/09/2018 10:54:02	Sala Farnese	13	id_0000001
09/09/2018 10:54:28	Ingresso	09/09/2018 10:54:15	Ingresso	13	id_0000001
09/09/2018 10:54:41	Atrio basso sinistro	09/09/2018 10:54:28	Ingresso	13	id_0000001
09/09/2018 10:54:53	Ingresso	09/09/2018 10:54:41	Atrio basso sinistro	12	id_0000001
09/09/2018 10:55:06	Ingresso	09/09/2018 10:54:53	Ingresso	13	id_0000001
09/09/2018 10:55:19	Ingresso	09/09/2018 10:55:06	Ingresso	13	id_0000001
09/09/2018 10:58:55	Ingresso	09/09/2018 10:55:19	Ingresso	216	id_0000001
09/09/2018 10:59:08	Atrio basso sinistro	09/09/2018 10:58:55	Ingresso	13	id_0000001
09/09/2018 10:59:20	Atrio alto destro	09/09/2018 10:59:08	Atrio basso sinistro	12	id_0000001
09/09/2018 10:59:33	Ingresso	09/09/2018 10:59:20	Atrio alto destro	13	id_0000001
09/09/2018 10:59:34	Atrio alto destro	09/09/2018 10:59:33	Ingresso	1	id_0000001
09/09/2018 10:59:59	Atrio alto destro	09/09/2018 10:59:34	Atrio alto destro	25	id_0000001
09/09/2018 11:00:12	Sala Mosaici	09/09/2018 10:59:59	Atrio alto destro	13	id_0000001
09/09/2018 11:00:25	Sala Mosaici	09/09/2018 11:00:12	Sala Mosaici	13	id_0000001
	Atrio alto destro	09/09/2018 11:00:25	Sala Mosaici	0	id_0000001
09/09/2018 11:00:38	Sala Mosaici	09/09/2018 11:00:25	Atrio alto destro	13	id_0000001
09/09/2018 11:00:50	Sala Mosaici	09/09/2018 11:00:38	Sala Mosaici	12	id_0000001

Path	E	H	U	F	B	C	G	Y	M	S	V	P	T	disp	total
EHCHBFHMSTSVSHE	470.0	1709.0	0.0	13.0	12.0	12.0	0.0	0.0	968.0	733.0	225.0	0.0	251.0	3252.0	7645.0
EHUFBHSVSPTSHE	208.0	372.0	88.0	176.0	188.0	0.0	0.0	0.0	0.0	442.0	113.0	63.0	352.0	1948.0	3950.0
EHPTSHCHUE	780.0	208.0	101.0	0.0	0.0	8.0	0.0	0.0	0.0	798.0	0.0	918.0	829.0	3026.0	6668.0
EHPTSHUE	858.0	42.0	113.0	0.0	0.0	0.0	0.0	0.0	0.0	798.0	0.0	1006.0	867.0	2984.0	6668.0
EHMVSTPTSHE	1327.0	246.0	0.0	0.0	0.0	0.0	0.0	0.0	1120.0	64.0	315.0	12.0	342.0	1022.0	4448.0
EFMHE	470.0	1919.0	0.0	605.0	0.0	0.0	0.0	0.0	9.0	0.0	0.0	0.0	0.0	302.0	3305.0
EHFHE	3766.0	1283.0	0.0	1516.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	231.0	6796.0
EHTBHSMVSHGHE	1694.0	621.0	0.0	0.0	401.0	0.0	101.0	0.0	922.0	2341.0	288.0	0.0	12.0	2091.0	8471.0
EHMHFHCHFBHFHFHFE	101.0	22335.0	0.0	2220.0	10.0	12.0	0.0	0.0	25.0	0.0	0.0	0.0	0.0	2880.0	27583.0
EHUBFHPTSMCHE	309.0	1992.0	189.0	12.0	163.0	25.0	0.0	0.0	403.0	242.0	0.0	127.0	632.0	1983.0	6077.0
EHBFHMVSTSHE	1123.0	1175.0	0.0	603.0	12.0	0.0	0.0	0.0	25.0	283.0	188.0	0.0	88.0	1662.0	5159.0
EHUFYBHMSPTSVHE	711.0	774.0	63.0	189.0	25.0	0.0	0.0	217.0	314.0	166.0	12.0	229.0	12.0	1559.0	4271.0
EHSPHE	603.0	2006.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	12.0	0.0	254.0	0.0	1919.0	4794.0
EHSPHBFHCHE	914.0	1164.0	0.0	277.0	86.0	12.0	0.0	0.0	0.0	12.0	0.0	254.0	0.0	2075.0	4794.0
EHBFHCHSHTPHTSHE	2.0	432.0	0.0	12.0	9.0	573.0	0.0	0.0	0.0	781.0	0.0	514.0	435.0	2053.0	4811.0
EHBFHFHFHCHFHFHFHE	8.0	1136.0	0.0	29250.0	12.0	508.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4963.0	35877.0
EHCHFHCHE	128.0	271.0	0.0	101.0	0.0	101.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	327.0	928.0
EHVSTPSMHFHE	6.0	1596.0	0.0	542.0	0.0	0.0	0.0	0.0	617.0	402.0	254.0	116.0	192.0	3426.0	7151.0
EHMTHFHE	458.0	1440.0	0.0	996.0	0.0	0.0	0.0	0.0	1318.0	0.0	0.0	0.0	379.0	3396.0	7987.0
EHMSPTSHE	89.0	2281.0	0.0	0.0	0.0	0.0	0.0	0.0	910.0	233.0	0.0	320.0	378.0	4279.0	8490.0
EHSTSPVHE	626.0	2182.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	63.0	693.0	778.0	76.0	3908.0	8326.0
EHMSTHE	486.0	2163.0	0.0	0.0	0.0	0.0	0.0	0.0	1104.0	265.0	0.0	0.0	217.0	2753.0	6988.0
EHFBHMSPTHE	48.0	1059.0	0.0	1324.0	428.0	0.0	0.0	0.0	1838.0	50.0	0.0	153.0	38.0	4783.0	9721.0
EHFHMSPTSVHE	12790.0	523.0	0.0	190.0	0.0	0.0	0.0	0.0	328.0	114.0	13.0	256.0	227.0	1936.0	16377.0
EHFHMSHE	567.0	2296.0	0.0	12.0	0.0	0.0	0.0	0.0	365.0	1614.0	0.0	0.0	0.0	1691.0	6545.0
EHCHCHBFHE	8.0	835.0	0.0	791.0	62.0	529.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	363.0	2588.0
EHE	10.0	3379.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	11.0	3400.0
EHFBHCVSTPSHMHE	63.0	585.0	0.0	315.0	214.0	88.0	0.0	0.0	591.0	853.0	466.0	969.0	190.0	3789.0	8123.0
EHE	123.0	8808.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	21.0	8952.0
EHBHE	6.0	1775.0	0.0	0.0	12.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	13874.0	15667.0
EHMVSTPFPHBHE	201.0	1548.0	0.0	3.0	50.0	0.0	0.0	0.0	1629.0	353.0	416.0	319.0	13.0	3770.0	8302.0
EHUFBHMSVMVMVTHCHE	2.0	1415.0	315.0	440.0	188.0	390.0	0.0	0.0	484.0	329.0	466.0	0.0	1356.0	3034.0	8419.0
EHCHE	382.0	101.0	0.0	0.0	0.0	431.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1977.0	2891.0
EHVSTPSMHBFHE	30.0	1666.0	0.0	290.0	12.0	0.0	0.0	0.0	757.0	300.0	519.0	91.0	229.0	3290.0	7184.0
EHVSTUHE	6.0	51.0	12.0	0.0	0.0	0.0	0.0	0.0	0.0	352.0	354.0	0.0	708.0	7704.0	9187.0
EHE	5043.0	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	11.0	5057.0
EHGHFHSPTSHE	490.0	1040.0	0.0	1397.0	0.0	0.0	453.0	0.0	0.0	987.0	0.0	306.0	25.0	2208.0	6906.0
EHUFHCVSPSHE	254.0	1332.0	253.0	778.0	0.0	12.0	0.0	0.0	0.0	1751.0	845.0	154.0	0.0	5384.0	10763.0
EHUFYGYBHCHVSPBHE	310.0	714.0	152.0	1205.0	224.0	329.0	252.0	303.0	0.0	1533.0	692.0	751.0	0.0	5100.0	11565.0
EHUFBHMSTSVCHE	2.0	898.0	63.0	189.0	63.0	330.0	0.0	0.0	365.0	116.0	25.0	0.0	13.0	2209.0	4273.0
EHUFYBHCHMSPSVHE	102.0	879.0	490.0	616.0	50.0	25.0	0.0	704.0	416.0	858.0	326.0	457.0	0.0	2367.0	7290.0
EHFBHSTPMHE	417.0	559.0	0.0	12.0	353.0	0.0	0.0	0.0	848.0	153.0	0.0	664.0	1045.0	4988.0	9039.0
EHUFHE	448.0	158.0	389.0	62.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	628.0	1685.0
EHMHCHBFBHCSHE	509.0	1120.0	0.0	140.0	25.0	138.0	0.0	0.0	366.0	1849.0	0.0	0.0	0.0	1255.0	5402.0
ECHE	2492.0	216.0	0.0	0.0	0.0	7901.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	132.0	10741.0
EHCE	2419.0	12.0	0.0	0.0	0.0	364.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	7668.0	10463.0

Figure 4.3: Sample log of the data. Sample of the processed and aggregate data.

The goal of the fourth stage is to split and segment the data, discover patterns, and extract features. Statistics can be used to discover significant factors. This is accomplished using basic statistics and the 15 features of real value data.

- The diagonal depicts the distribution of each variable.
- The bivariate scatter plots with a fitted line are shown at the bottom of the diagonal.
- The correlation value and significance level are shown as stars at the top of

the diagonal.

- A symbol is assigned to each significance level: p-values(0.001, 0.01, 0.05, 0.1, 1) are matched with symbols("***", "**", "*", ".", " ")

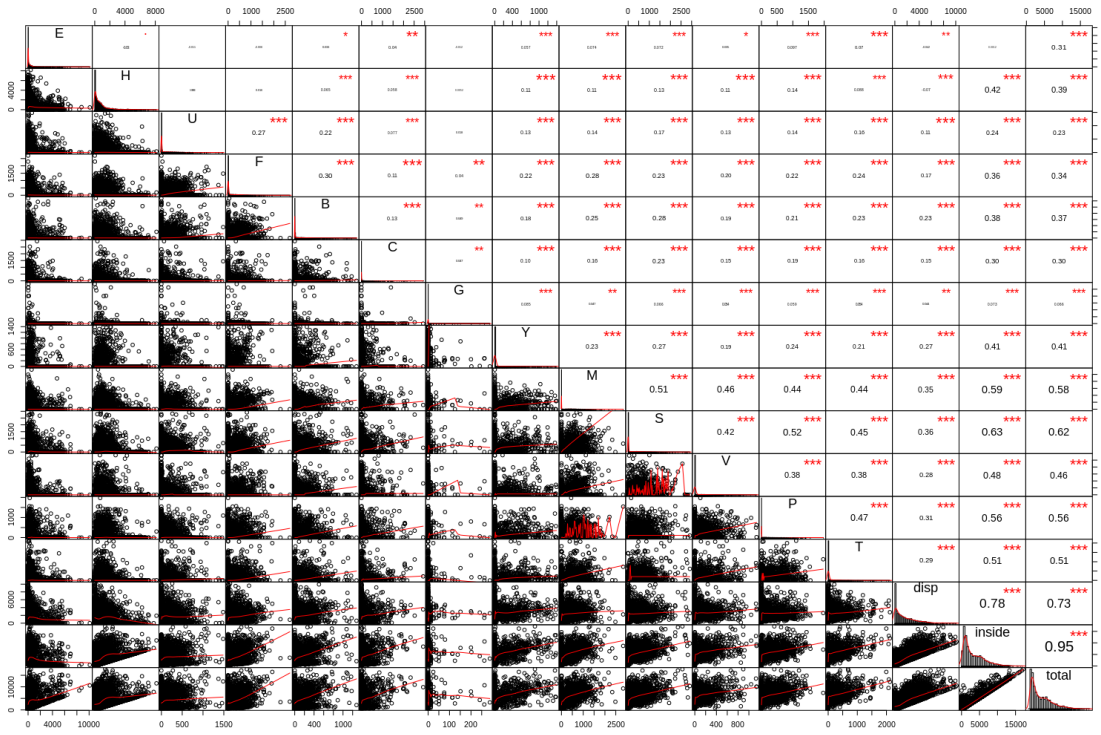


Figure 4.4: A matrix of correlation plots illustrating the relationships between pairs of variables. Variable histograms are displayed along the diagonal of the matrix; scatter plots of variable pairs are displayed in the lower diagonal. On the top diagonal, the correlation coefficient and significance level are denoted by stars.

A combinatorial problem In our dataset

- $n = 19000$ unique users behavioural data
- $d = 16$ features (1 Path + 15 δ-time) for each data point $\mathbf{x}_i$
- $K = 8$
- Number of possible partitions $\mathbf{z}$: $\#\{\mathbf{z}\} \approx \frac{K^n}{K!}$

- For instance: $\#\{\mathbf{z}\} \approx 10^{17154}$

X denotes the entirety of the dataset. In our scenario, n=20,000 visits, and each data point is identified by d=16 attributes, one of which is a string type (paht), while the remaining 15 are time values. If we assume K = 8 clusters, this is the full set of potential combinations. The primary objective of our case study is to determine the lengths between strings. We investigate four types of distances: two are based on q-gram similarity, while the other two are based on edit similarity.

Example of paths:

EHE

EHBFHUHE

EHCHE

EHCHBHE

EHUFBHMTPSHE

EHFHE

EFMHE

EHMHE

Similarity and distance functions

Cosine similarity We have the cosine similarity where each string is vectorized, as the number of occurences och each letter as the components of the vector. As the modeal bag of words for each letter. Here we see an excample, the cos ranges from 0 to 1, since we are counting values can be only positibe, so the anlge is only betrein 0 and 90 degrees. Next we have the repserntation of the space, in our case study, with all the data set of octrober, the color in the level prol is generally cleares, so they hare colse to each other, Nad the clustering is not so good, since there are many data point non well suited in the clsuter, since some siluett point is less than zero, and the avrage is 0.3

$$s_{cos}(S,T) = \frac{v(S) \cdot v(T)}{\|v(S)\|_2 \, \|v(T)\|_2}$$

where $v(\cdot)$ is a nonnegative integer vector which components represents the number of occurrences of every possible element of the string.

For example: $\mathbf{a} = EHFHE$

$\mathbf{b} = EHE$

$\mathbf{c} = EHUFBHMTPSHE$

$\delta_{cos}(\mathbf{a}, \mathbf{b}) = 0.11$

$\delta_{cos}(\mathbf{a}, \mathbf{c}) = 0.18$

$\delta_{cos}(\mathbf{b}, \mathbf{c}) = 0.30$

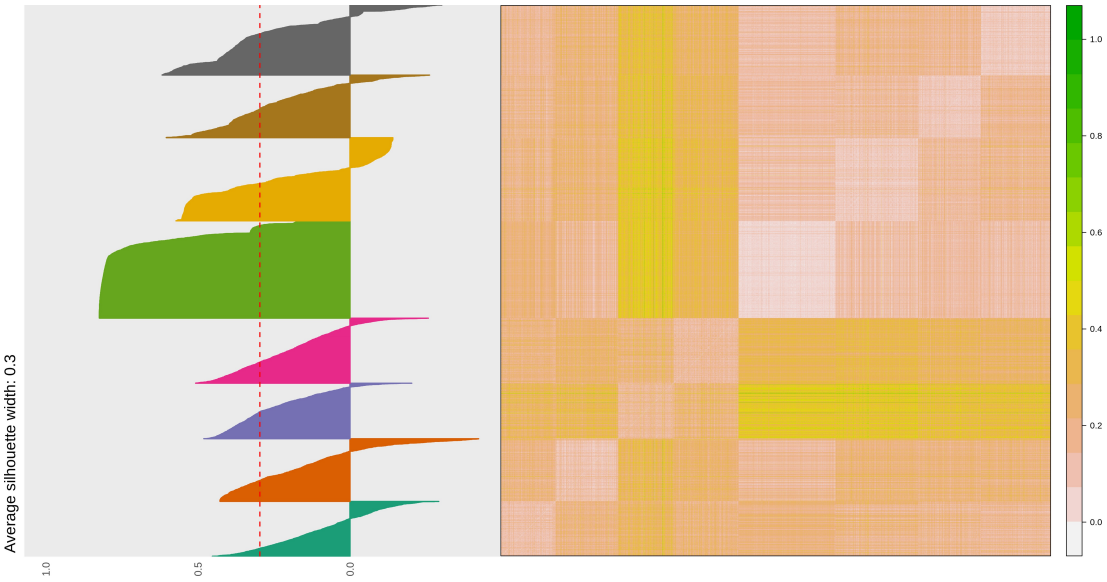


Figure 4.5: As an example, the cos ranges from 0 to 1, and because we are counting only positive numbers, the angle is limited to between 0 and 90 degrees. Following that, we have the representation of the space; in our case study, with all the octrober data sets, the color in the level prol is generally clear, indicating that they are adjacent to one another. However, the clustering is not optimal, as there are many data points that are not well suited to the clsuter, as some siluett points are less than zero, and the average is 0.3.

Jaccard similarity

$$s_{\text{Jac}}(S,T) = \frac{|Q(S) \cap Q(T)|}{|Q(S) \cup Q(T)|}$$

where $Q(\cdot)$ is the unique set of every occurring element of the string.

Here we have a more accurate siluette index, 0,4 with Jaccard, and we are looking at single occurrences rather than the number of times I was in a node. Thus, we examine the set of single letters included within a string. From the level porl, we can see that the data is widely dispersed and that the range between 0 and 1 is pretty large.

These two analogies are based on q-grams, in which we examine the singular occurrences of the letter in a single gram.

Example:

$\mathbf{a} = EHFHE$

$\mathbf{b} = EHE$

$\mathbf{c} = EHUFBHMTPSHE$

$\delta_{Jac}(\mathbf{a},\mathbf{b}) = 1 - \frac{2}{3} = 0.33$

$\delta_{Jac}(\mathbf{a},\mathbf{c}) = 1 - \frac{1}{3} = 0.67$

$\delta_{Jac}(\mathbf{b},\mathbf{c}) = 1 - \frac{2}{9} = 0.78$

Longest Common Substring Next we pass to edit base types, LCS. Here we look at the edit operation, in this case this operations are only insertion or deletion operations.

$\delta_{LCS}(S,T) = \frac{LCS(|S|,|T|)}{|S|+|T|}$ where

$$LCS(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ LCS(i-1,j-1) & \text{if } S(i) = T(j), \\ 1 + \min\{LCS(i-1,j), LCS(i,j-1)\} & \textit{otherwise.} \end{cases}$$

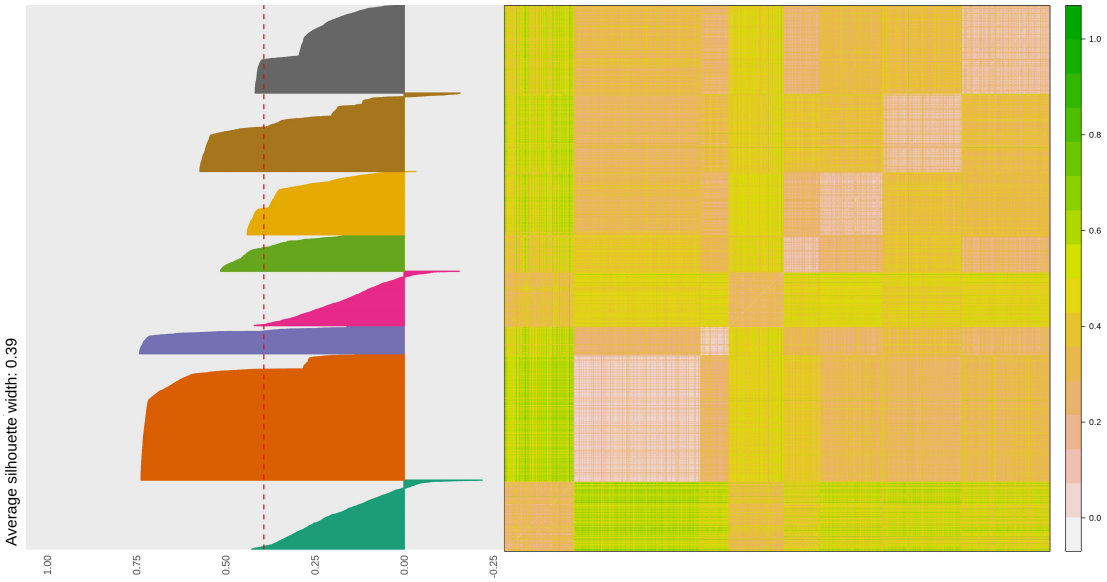


Figure 4.6: Here we have a more accurate siluette index, 0,4 with Jaccard, and we are looking at single occurrences rather than the number of times I was in a node. Thus, we examine the set of single letters included within a string. From the lavel porl, we can see that the data is widely dispersed and that the range between 0 and 1 is pretty large.

Example: $\mathbf{a} = EHFHE$

$\mathbf{b} = EHE$

$\mathbf{c} = EHUFBHMTPSHE$

$\delta_{LCS}(\mathbf{a}, \mathbf{b}) = \frac{2}{8} = 0.25$

$\delta_{LCS}(\mathbf{a}, \mathbf{c}) = \frac{7}{17} = 0.41$

$\delta_{LCS}(\mathbf{b}, \mathbf{c}) = \frac{9}{15} = 0.60$

Levenshtein Distance In the LV, we have edit distances, but not only insertion and deletion but also substitution.

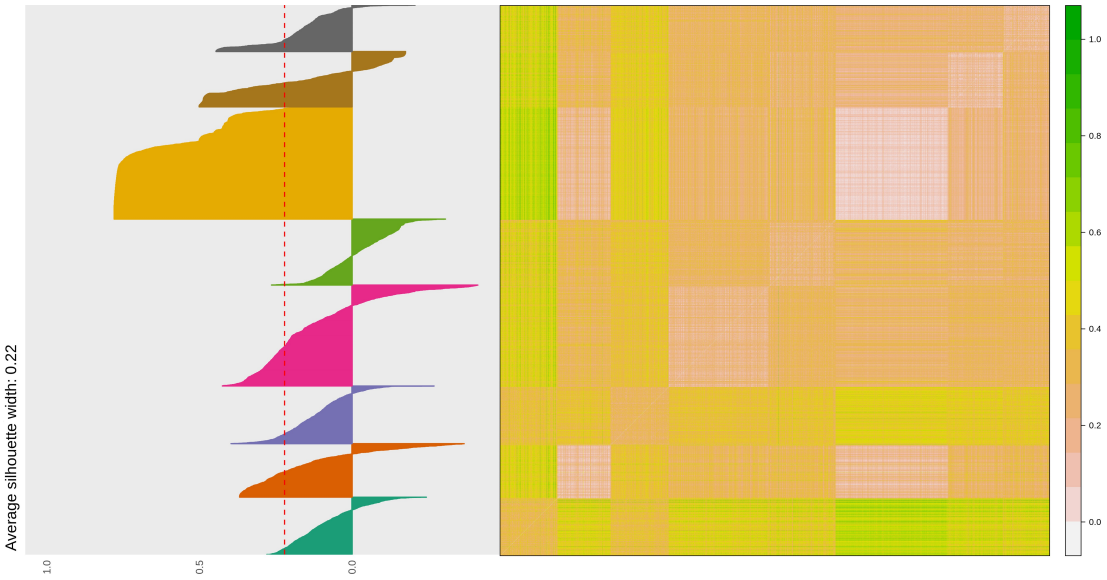


Figure 4.7: LCS

$\delta_{LD}(S,T) = \frac{LD(|S|,|T|)}{\max\{|S|,|T|\}}$ where

$$LD(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} LD(i-1,j)+1 \\ LD(i,j-1)+1 \\ LD(i-1,j-1)+1_{(S_i \neq T_j)} \end{cases} & \text{otherwise.} \end{cases}$$

Example: $\mathbf{a} = EHFHE$

$\mathbf{b} = EHE$

$\mathbf{c} = EHUFBHMTPSHE$

$\delta_{LD}(\mathbf{a},\mathbf{b}) = \frac{2}{5} = 0.40$

$\delta_{LD}(\mathbf{a},\mathbf{c}) = \frac{7}{12} = 0.58$

$\delta_{LD}(\mathbf{b},\mathbf{c}) = \frac{9}{12} = 0.75$

Accuracy indices for clustering He wishes to compare the results of clustering. We can do a comparison through the use of the F-measure, obtained by the harmonic mean of precision and recall. Which is derived from the Confusion matrix. We examine which data belongs to which cluster and which data is missing.

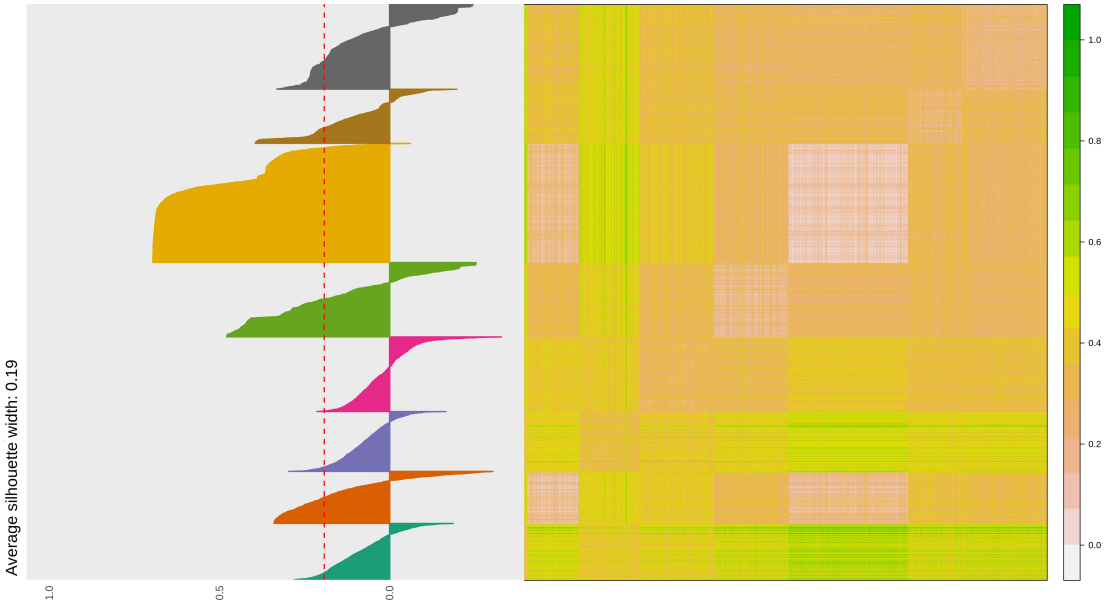


Figure 4.8: LV

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Figure 4.9: Confusion matrix

$$\textit{Precision \& Recall: } P = \frac{TP}{TP+FP} \qquad R = \frac{TP}{TP+FN}$$

$$\textit{F-measure: } F = \frac{2PR}{P+R} = \frac{2}{\frac{1}{P}+\frac{1}{R}}$$

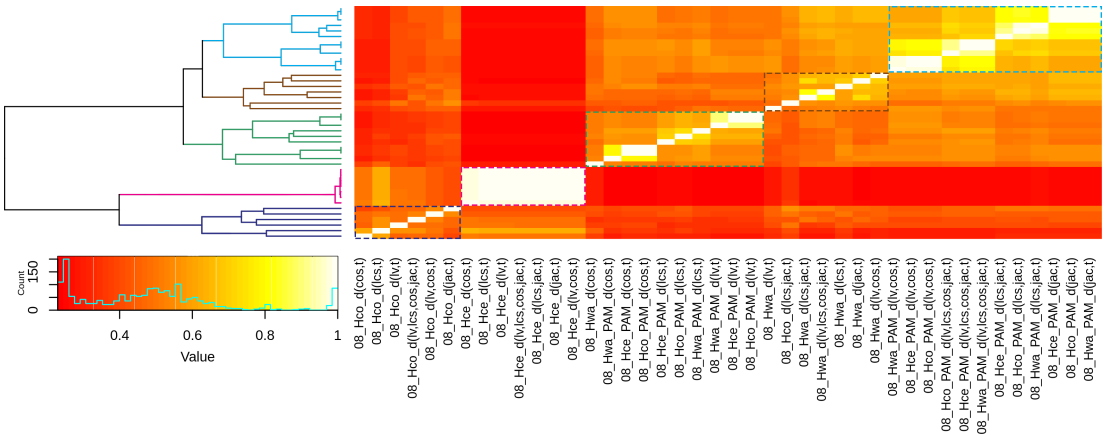


Figure 4.10: Ordered F-measure heatmap with comparison among all the experimented clustering methodologies.

Comparison This study was extended to multiple combinations of clustering techniques and different sum of distances.

Castel Nuovo

The second museum of our interest is the *Museo Civico of Castel Nuovo* in Naples, located inside *Maschio Angoino* castle containing various artworks from a wide range of historical time periods. They are organized into four floors, as it can be seen in Fig. 4.11, which has been presented in [105;106;107].

During the data collection period, which spans from May 2017 to October 2017, each visitor had the opportunity to rent a mobile interactive device for an additional fee; 4719 visitors in total agreed to rent and use it. Thanks to this devices, visitors could select among over than 160 cultural items with multimedia facilities.

The mobile device incorporates a customized application, which has been developed for both the purpose of giving additional information to the visitor through augmented reality (audio files, photos, text for each artwork) and tracking

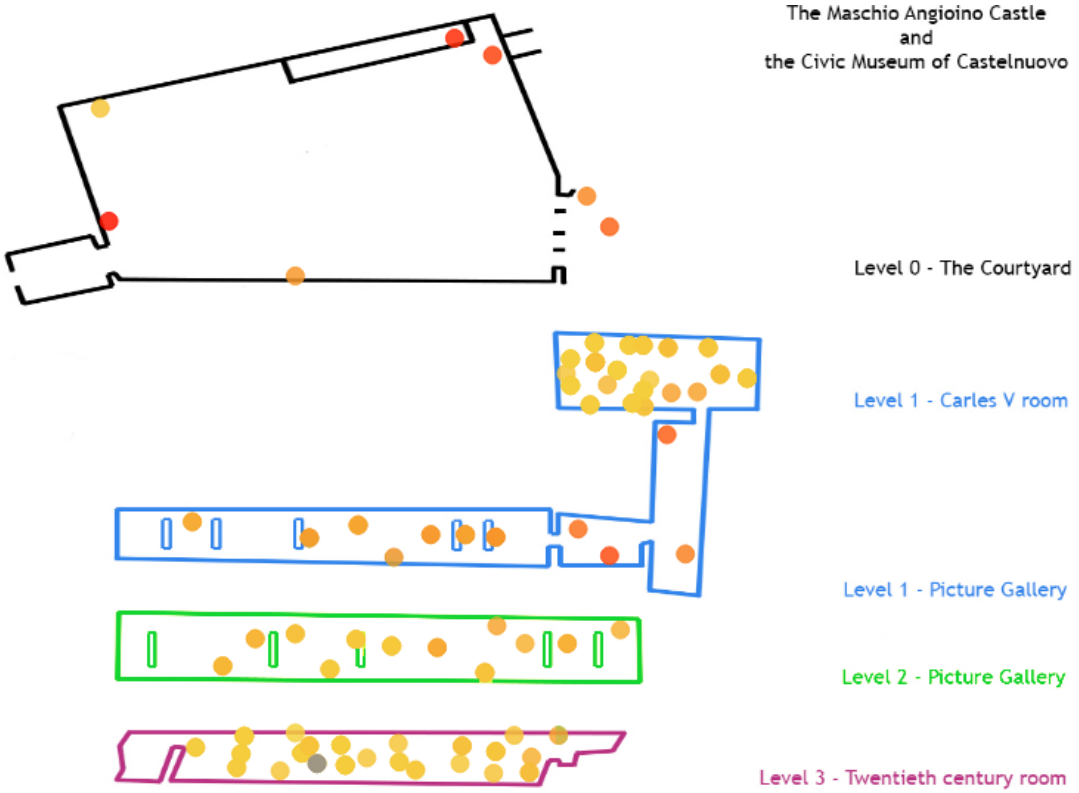


Figure 4.11: Floor plan of *Museo Civico del Castel Nuovo* in Naples. The dots in the map represent the position of the beacons; the color of each dot indicates the number of visitors interacting with the beacon. Yellow, orange and red color indicates low, medium and high numbers of visitors respectively.

the actions of the visitor. All the aspects and some more actions are collected as records in the dataset, and this procedure is also thoroughly described in [105;107].

This section briefly describes how data have been elaborated as input for the learning procedure. As it has been mentioned before, we collected the data during a 5 months period. After an initial preprocessing, more than 30 features have been extracted. Then, only the multimedia features related to the artworks have been selected, resulting in 19 features in total. The list and the description of these are presented in Table 4.1. The first two features (`lang` and `month`) are categorical. Other features related to time-dependent variables, number of actions on the device (that can be either audio, photo or text related).

The information profiling step from the selected features has been performed with an *unsupervised learning* methodology. Differently from the approach in [105], which used the K-means algorithm, in this paper we applied a K-medoids algorithm to the distance matrix obtained only by quantitative features. Before proceeding with the clustering, since some features can have different units of measure, a normalization step has been performed. In details, a partition of the features is performed by the same unit of measure; then, a distance matrix is computed for each group by means of the *Euclidean distance*. All resulting matrices have been merged into one matrix with the following formula also used in [52;101] and reported here:

$$D_{\text{tot}} = \sum_{i=1}^{G} \omega_i D^{(i)} \tag{4.1}$$

where $D^{(i)}$ is the distance matrix for the i-th group, G is the number of groups and ω_i is the inverse of the maximum element in $D^{(i)}$. After calculating the distance matrix D_{tot}, a clustering step is performed by using the PAM algorithm [61;111]. For the implementation point of view, the `R` package `cluster` [79] has been adopted.

Field	Content
lang	Favorite language chosen in the tablet
month	month of the visit
n_audio_full	Distinct count of artworks whose audio files were played in full by the visitor
ratio_n_artwork	Distinct count of artworks visited, percentage with respect the maximum (169)
sec_visit	Time elapsed from the first artwork selection to the end of the tablet-based visit, measured in seconds
sec_hole	hole time, sum of all the times between the last log action on an artwork, and the selection of the next artwork.
n_text_invisible	No. of actions with text_button = invisible
n_text_visible	No. of actions with text_button = visible
n_audio_stop	No. of actions which interrupt listening of the audio track (e.g. play_button_playing = false)
n_audio_active	No. of actions which explicitly keep active the audio track (e.g. play_button_playing = true)
n_audio_seektrack	No. of seek actions on the audio track bar.
ratio_audio_full	Ratio between the number of fully listened audio and the number of visited artworks.
sec_action	Time of the usage of the device (calculated by sec_visiti - sec_hole)
n_total_operations	Number of tablet-based actions performed by the visitor on artworks
n_photo_action	No. of actions on photos/pictures (nextpic_photo, prevpic_photo)
ratio_artwork_F*	No. of visited artwork for each floor (where * can be 0,1,2,3)

Table 4.1: The dataset features

4.1.2 e-health

KG applications

Among KG application related to health care booking systems, to the best of the authors' knowledge, some research can be found, although not combining all these aspects. KGE can be used for co-author recommendation[62], by proposing a pipeline that creates a KG, then an embedding step, and finally a Metaresearch Recommendation. It is important to remark that EHRs are the primary source of data related to patient health in a digital format containing medical knowledge[118;135]. The authors in[56] proposed a methodology for generating a knowledge network (MKN) related to medical diagnosis by means of Chinese EHRs. Learning from EHRs, casual relationships involving diseases and symptoms have been discussed in[124] where the authors propose a KG approach through a clinical Bayesian network. Starting from the structured and unstructured data of patients, the authors in[80] proposed an automated KG construction related to cerebral aneurysms disease. The potential of the graph pattern feature-based approaches in KG have been discussed by the authors in[9] in order to predict treatment and causative relations. Additionally, the authors in[71] explored the potential of KG to predict biomedical relationships, also considering the graph features to help with link predictions. Considering the classification problem of diseases within the framework of e-health and EHR data, the authors in[69] discussed a data-driven approach based on ML and KG. This idea could be extended to manage the booking process, from a managerial perspective, in depicting and scrutinizing the overall links that appear from these processes.

Contributions In the framework of this related work, KG applications Embedding procedures are at the core of information extractions directly from KG. Moreover, a database that can support complex knowledge modelling should be able to reduce the implementation time of such pipeline.

The main contributions of this thesis are as follows:

- The design of a KG approach, through a pipeline, supporting the healthcare domain using real-world booking data;
- A novel intelligent framework (and database) for the KG, named GRAKN, which has been presented and also used to model a part of the healthcare domain;
- The exploitation of the Knowledge Graph Embedding to apply Machine Learning techniques and provide interesting insights regarding the data.

CUP dataset

Referrals and medical services are accessed through a Booking Centre managed by local health authorities and controlled by regional governments in Italy's public healthcare system. We shall provide our research projects on Health care booking data. How we passed from a legacy dataset established in the 1980s, and how we sought to extract insights with AI solutions. The data sent to our research laboratory Modal by an IT company that maintains the booking data of the Local Health Department of Naples was chaotic. The organization gathered years of data, saved it in their databases, spread it throughout the region, and faced various revisions and changes throughout the years. They questioned if we thought we'd be able to locate anything interesting in all that data. Data from patients who booked a health service with a prescription after being referred by their practitioner.

The idea was to apply AI solutions to legacy data, i.e. extract insights with machine learning from that data, construct a knowledge graph from tuple data, and since we started with a legacy SQL-like database, have been utilized as a bunch of data, with no Entity-relation structure. Prediction and analytics in real time. Why did we wish to take this approach? Because booking processes are extensively utilized, our approach can be replicated in a variety of additional applications and

data. Since we have booking medical prescriptions, this data may be utilized to anticipate what type of health care a patient may require, as well as estimate the load and KPI of health service providers, by modifying the predicting subject's point of view. Medical data is semantic, which means that it has a meaning in terms of the medical domain the health care belongs to. This resulted an issues. Challenges include the need to swiftly create a knowledge base (define a schema, load data, and so on). Transition to sophisticated knowledge graph applications (entity resolution, link prediction, pattern discovery) brought to use ML on KGs.

Booking procedure explanation We have a patient that goes to her General Practitioner, who examines her and determines which health services she requires. The GP orders the health service, such as an eco-doppler, an electrocardiogram, a blood test, or surgery. Health services under public healthcare are obtained through a Booking Centre run by local health agencies and overseen by regional governments. As a result, the patient makes a booking at the Booking Centre for the health service. Perhaps the patient will rebook, change the date, and eventually pay.

Paradigm We presented this paradigm, which is organized as a task pipeline, summarized in Figure 4.12.

It consists of five steps:

- in order to address the challenge of structuring knowledge from databases, we had to do a pre-processing phase of cleansing and knowledge base schema extraction, which included extracting entities, attributes, meta-relationships, and so on. In this regard, our strategy combines data-driven and domain knowledge-driven methodologies.
- a phase for storing data in a graph database
- a phase of graph data embedding, followed by

- a phase of machine learning on real-valued data

The methodology enables a company to apply artificial intelligence to previously owned tabular data.

So this framework approach is generalized, and it has been created to assist an organization, such as (in the example we are providing) a public healthcare organization, in transitioning from traditional SQL-like data administration to a Knowledge Graph for Machine Learning insights extraction.

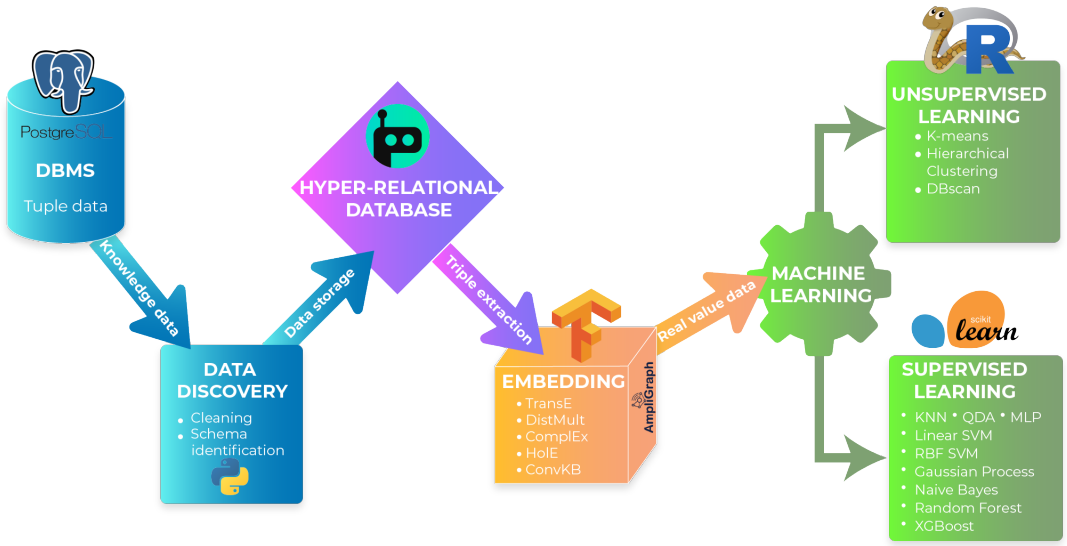


Figure 4.12: Visual pipeline for processing data from and SQL for applying machine learning to data in a Knowledge Graph form

4.1.3 Knowledge Graph for CUP

This section presents and describes a KG framework operating on e-health data to manage semantic concepts and relationships. The proposed framework has been structured as a pipeline of tasks summarized in Figure 4.13. It is composed of (i) a preprocessing phase for data cleaning and knowledge base schema extraction, (ii) a data storing phase in a graph database, (iii) a graph data embedding phase, and finally (iv) an ML phase on real-valued data. Insights about each step will be described in the next sub-sections, also with their application in relation to e-health

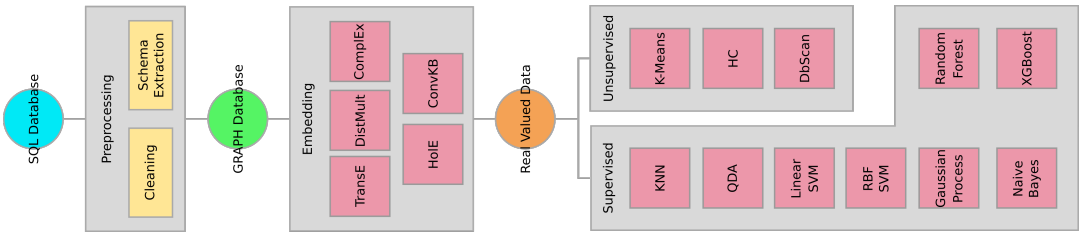


Figure 4.13: Pipeline diagram. This diagram shows our architecture, including all its steps. From a SQL database, we extract data on which we create a knowledge base schema, thanks to business domain knowledge and data preprocessing which can involve cleaning and data-driven ontology creation. Once the schema has been generated, a KG database is populated from the SQL data, according to the schema. Afterwards, with the triples in the Graph database, an embedding process is executed to obtain real value data. Finally, such real value data are used to give insights, using ML techniques and predictions, together with data visualization.

data. Our methodology has been designed to assist in terms of organization, e.g. to help a public healthcare system shift from a classic Structured Query Language (SQL)-like data management to a KG for ML insights extraction. The methodology enables an organization to apply Artificial Intelligence (AI) to previously owned tabular data. In the result section, we will show the application of our framework to a real-world example and discuss all the available results supporting stakeholders and decision-makers.

Entity-relations extraction

Starting from the first step of Figure 4.13, entity-relations extraction represents the transition from an SQL database to a *semantic* structure of data. Recognizing entities and relations from tabular data is a challenging task. This problem is also known as KGI.Our proposed approach is a combination of data-driven and domain knowledge-driven approaches. In such a case, the data are not structured into an entity-relation database and are in a tabular form. An in-depth study of the relationships between pairs of rows sharing the same values on one or more fields is

performed. Indeed, when rows in a table share the same value on a fixed field, they may be linked to the same relation or entity. Entity disambiguation is performed by counting the unique occurrences of couples of columns in the database in order to obtain an overall view of which pairs of fields can be considered as belonging or not to an entity or relation. Finally, in our case, this process was refined with some domain experts, to obtain the final KG schema.

Concerning the data cleaning process, it has to be taken into account that our tabular data also had indexes and descriptions that, although theoretically standardized, had sometimes been manually modified by human operators. We therefore designed and implemented a Python procedure that performed a string matching in order to cluster similar strings, whose differences could have been generated by spelling errors, naming convention changes or text contraction. Finally, we added flag-columns indicating the quality of each row of the database, to be used in further analysis.

Graph database

Once the data have been organized and cleaned, and a *semantic* structure has been defined, they have to be transferred to the second step of Figure 4.13, the data storing in a graph database.

In order to achieve improvements with graph structures and their extensions to a knowledge structure related to our business domain, an intelligent tool is required. We have preliminary analyzed different graph databases, such as Neo4j, OrientDB, JanusGraph (originally Titan), Grakn, GraphDB (formerly OWLIM), finally selecting Grakn. Our research has been focused on finding a KG database that would quickly help us to build a knowledge base, so as to shift to KG complex applications, such as entity resolution, link prediction and pattern discovery. We have also been interested in analyzing all possible solutions, also taking into account newer and less well-known ones. While Grakn was first released in 2016, most

of the best-known graph databases had initially been released between 2000 (like GraphDB) and 2012 (like Titan) (for example, Neo4j was first released in 2007). In our opinion, Grakn seems the most suitable tool for building a KG, a choice which has also been supported by recent works. We could not find any similar solution within other databases.
Finally, although Grakn, is one of the most recent solutions within the graph database realm, it represents a very suitable solution to create and manage knowledge graphs.

We investigated many solutions and applications, including as Neo4j, OrientDB, and JanusGraph. Grakn appeared to us to be the best tool for creating a knowledge graph, and this choice is reinforced by recent work. Furthermore, Shinavier sees in Grakn a suitable data model for hypergraph database that addresses the challenge of aligning graph and relational databases, coherently manages hyperelements, i.e. generalized relationships, as described in their work in a 2019 algebraic properties paper.

Embedding and Machine Learning

Given a KG populated with its data, the process of gathering insights can start. First, an embedding phase has to be performed to obtain real value data the ML algorithms. In our framework, we have selected AmpliGraph embedding[31] due to its quality, updates, support and open-source code availability.

ML techniques are the principal methods for the extraction of insights from real-valued data. Previous research experiences related to unsupervised learning can be found in[23;106]. To obtain useful insights, starting from the embedding results we selected the number of clusters, k, with a heuristic approach based on the suggested quality cluster indexes implemented in the well-known `NbClust` package (see[25]), customizing the final selection. In more detail, we applied the principles of the techniques discussed in[140] to find a prominent elbow suggesting the most

appropriate number of existing clusters inside the data, i.e. *difference-like criteria*. In particular, we looked at each triple $(k-1, k, k+1)$ to find the local maximum, and finally, considering all these maxima or prominent elbows and different quality cluster indexes, we selected the final k with an ensemble approach through a voting technique. Once k had been chosen, we applied different clustering techniques, such as the K-means, Hierarchical Clustering[109] and DBscan[100] algorithms. Furthermore, we also exploited supervised techniques to build up a prediction step on data. The following algorithms were used and compared: KNN, QDA, MLP, Linear SVM, RBF SVM, Gaussian Process, Naive Bayes, Random Forest, and XGBoost, from the scikit-learn Python library[100].

We need to use Machine Learning to extract information from this data. We spend a significant amount of effort cleansing this data and ensuring that it is free of duplicates, mistakes, and so on. What we want to do is try to convert them into structured data that can be used in our machine learning pipeline. We then integrate this data into our unsupervised and supervised learning systems. In reality, we will want to avoid the feature engineering phase because it will make our lives lot easier. So, basically, we want to learn the features automatically. This is known as embedding.

Triples extraction To apply the Embedding to a Knowledge Graph, we must have all of the Knowledge Graph's triples. A triple is made up of two entities and one relationship. These are distinct from Grakn's entities and relations. For instance, (one patient, has-patient-age, 30) and (one referral, is-made-by, one doctor). To retrieve all of these triples, we reverse engineered Grakn. For example, for entities, we ran a query to get all of them and then looked at their attributes to see what is connected. To the right is a sorted table containing the number of extracted triples based on the specific relation that expresses the property of either an entity or a relation. For example, given the @has-patient-age relation, we extracted 32 thousand triples. Finally, we receive 1.5 million triples in total.

Unsupervised Learning on embedded space We want to use a 300-dimensional space embedded, fomr KG, to apply various unsupervised learning techniques such as K-Means, Hierarchical Clustering, and DbScan to find useful and hidden groups of different entities in order to retrieve non-trivial insights from the Knowledge Graph because, thanks to Knowledge Graph Embedding, we have one space containing all the entities. As a result, identifying hidden relationships between such organizations can constitute a value-added for stakeholders. Unfortunately, we tried to locate appropriate hyperparameters with DbScan using a random grid search on over 10k (thousand) permutations, however the results examined with the Silhouette index were unimpressive. Instead, the initial step in K-Means and Hierarchical Clustering is to determine the number of clusters, or the k-value.

Chapter 5

Smart cities scenarios

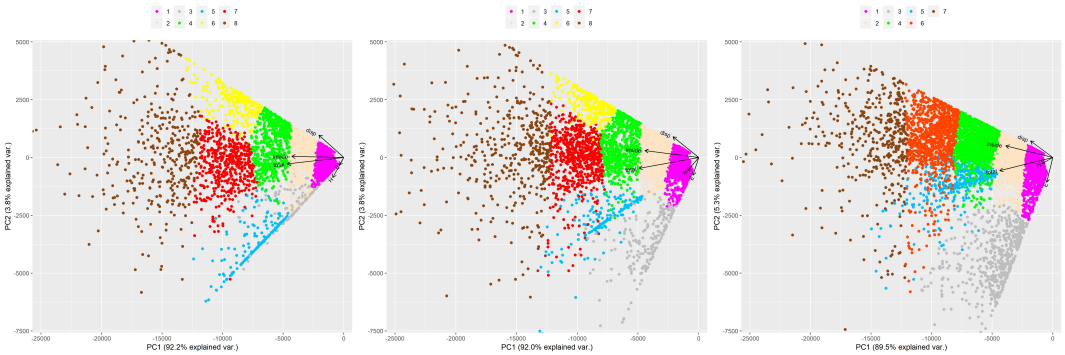
This chapter discusses and presents my findings. What are the findings of my research and how do they connect to earlier studies. Finally, this chapter presents is the pattern and categorization that emerges from the data and my findings and how they connect to the literature review and theoretical framework discussed in the previous chapter.

5.1 Museum Experiments

The main goal of this research study is an in-depth investigation of the obtained unsupervised classifications results to support situation awareness and decision-making processes for museum stakeholders. We consider two case-of-study with different IoT collected behavioral data and decision-making perspectives.

5.1.1 The M.A.N.N. results

According to the previously described heuristic approach, the best number of clusters turned out to be $K = 8$ for the months of August and September, and $K = 7$ for the month of October.



(a) AUG (b) SEP (c) OCT

Figure 5.1: PCA in the 3 months. PCA visualization, varying by months. Labels are the results of the PAM clustering technique. These three figures represent the data projected in the 2D space of the principal components. Arrows show the main features of the dataset projected onto the PCA space, so their lengths represent the variability of the features itself. It can be observed that the red and yellow clusters merge together becoming one single cluster in the month of October, as well as that the grey cluster spreads out over time.

Principal Component Analysis How clustered people's behaviors vary by month can be observed through the application of the Principal Component Analysis (PCA). This technique allows the analysis of data in a lower-dimensional space, whose axis represents the maximum variability directions, called principal components. In Fig. 5.1 a two-dimensional PCA is shown, where the principal axis accounts for as much of the variability in the data as possible. Considering the Fig. 5.1 it can be observed that for every month, the most important directions (features), are: the *displacement* time, the *inside* time, the *total* time, the time in node E (the *entrance*/*exit* node), and the time in H (the main *hall* of the museum after the entrance node). The angle between the E axis and the others increases month-by-month; this can be noticed especially by the detachment of the grey dots from the blue points. Another important observation is the merging of the yellow and red clusters into the orange cluster, which appears in October, because of the lack of separation among them; this aspect will be more in-depth analyzed in the next section.

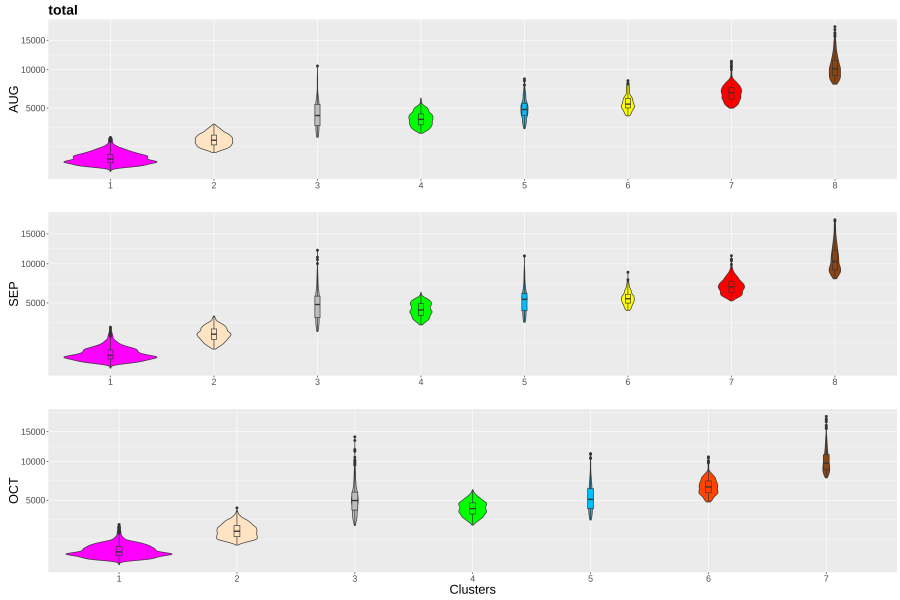
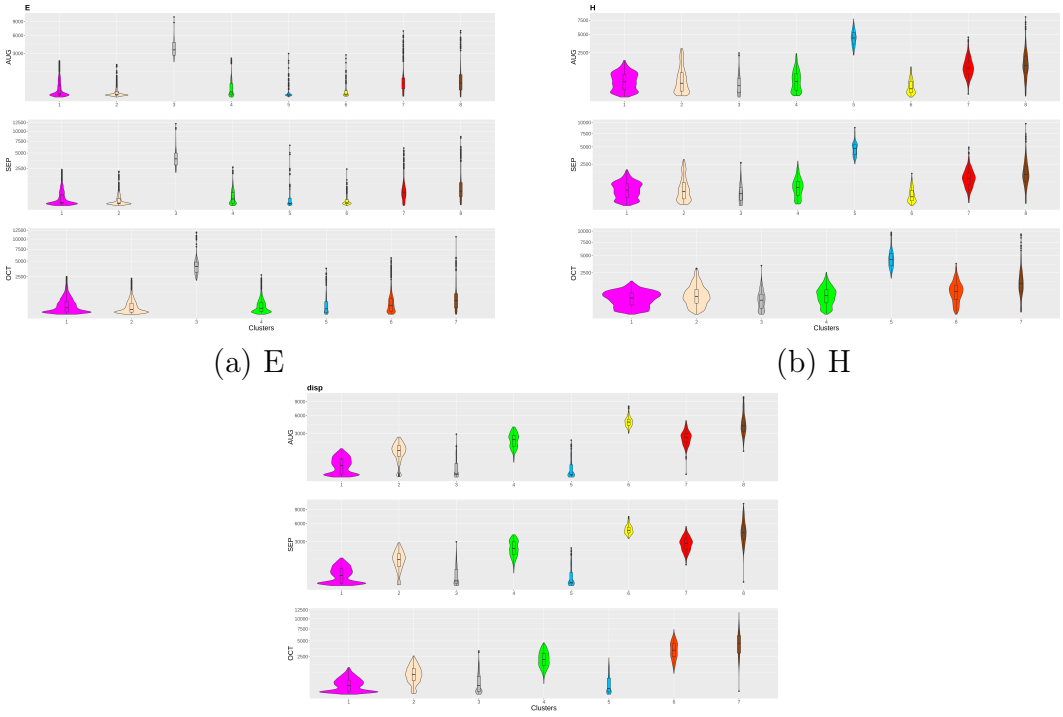


Figure 5.2: Monthly cluster profiles results, obtained by K-Medoids (PAM)

Violin plot of the clustering results profile[61].

Clusters' semantics

In order to analyze the *semantics* of the clusters and to propose a useful meaning of them, and finally, to produce useful insights for context-aware and decision-making processes, some considerations can be made by observing the results in Fig. 5.4. The best way to consider these results is to follow the most distinctive features appearing in the PCA. Since the total time turns out to be the main feature, the obtained clusters are mainly ordered by such feature. Violet and beige clusters encompass the visitors that are less engaged; they strongly increase in October getting relatively big with respect to the other clusters in the same month. The Green cluster represents the visitors that stay at the ground floor, and that mildly look around on the first level of the museum. Gray and Blue clusters represent the visitors that stay less time in any node of the museum with respect to the other visitors and are also quite well tracked. Gray visitors stay more in the node E, while blue visitors stay mostly in the node H. In detail, the Gray cluster represents



(a) E

(b) H

(c) disp

Figure 5.3: Monthly cluster profiles results, obtained by K-Medoids (PAM)

the visitors who stayed more in *E* and did not visit any of the *internal* areas of the museum. The Blue cluster represents the extemporary visitors, i.e. the visitors who have been most of their time in the main hall *H* of the museum. The last 2/3 clusters, i.e. yellow/red (which become orange in October) and brown are the ones that represent visitors staying more time in the museum. The Brown cluster represents extremely involved visitors. Fig. 5.4 highlights that yellow and red clusters merge together into the orange cluster in October. The Yellow cluster also denotes visitors who spent more time in the courtyard of the museum (where no IoT devices are located).

From a situational awareness and decision making perspective, this dataset, considering a month-by-month analysis of the visits, shows an increasing number of visitors with a change in their behavior. On the one hand, the data shows a growth of visitors whose interest in the museum visit is low (violet and beige clusters).

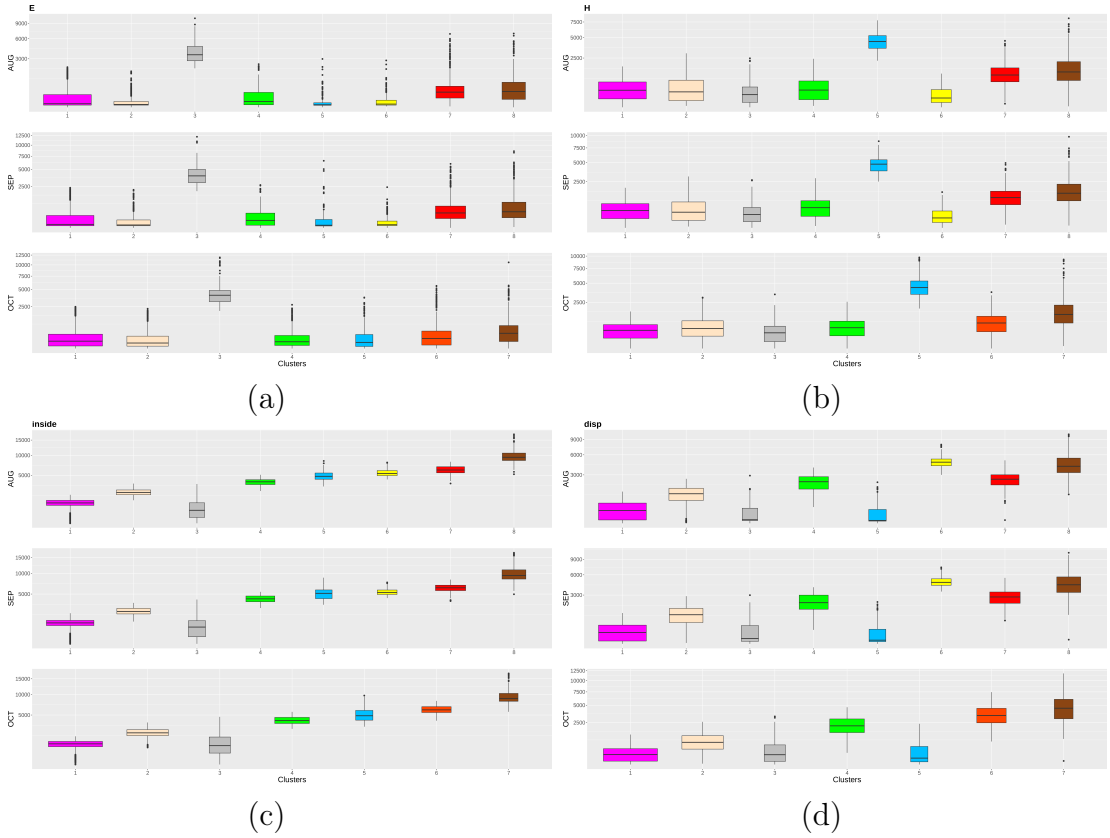


Figure 5.4: Grid of boxplots, describing for the main feature the composition among the different clusters. It has been used as a square root scale for the y-axis.

Furthermore, the clusters of more engaged people decrease in percentage within the month. It can be also observed that the gray cluster is increasing in percentage size with respect to the monthly number of visitors, and it is increasing in (mean) waiting time.

Finally, from a context-based observation point of view, it can be observed a reduction of the percentage of visitors in node S (the main hall on the second floor, *The Hall of the Sundial*), in the month of October, even though in the same month, the number of total visitors is almost doubled.

All these extracted useful information through unsupervised learning techniques on IoT data can provide and assess a starting point for a context-aware/decision-making system also providing real-time insights. The obtained clusters are also helpful for labeling the dataset and recognizing future visitors' behaviors through

supervised classification algorithms.

Clustering Results

In this section, we perform and discuss a comparison between the clustering results by giving an overall view of them. To measure the effectiveness of the clustering, the two most frequent measures are *precision* and *recall*[67]. These are defined as follows:

- *Precision*:
$$p = \frac{TP}{TP + FP}$$
- *Recall*:
$$r = \frac{TP}{TP + FN}$$

where TP are true positives, FP false positives and FN false negatives. Unfortunately these measures singularly are not appropriate. Precision and Recall are complementary measures, since high value of Precision usually correspond to low value of Recall[81]. A solution is to use the *F1 Score*, as used in[58]:

$$F_1 = 2 \cdot \frac{p \cdot r}{p + r}$$

Such score is also known as the Sørensen-Dice coefficient[38;129]. It is the harmonic average of precision and recall, so high values of *F1 Score* means that precision and recall are both high, and it is also symmetric. Authors in[67] advise to choose the *F1 Score* because it is *class-specific*.

Figure 5.5 reports all the *F1* measures calculated by the combination of all the distances and clustering methods. From this figure, it can be seen that an intersection value of 0.67 between the two Hierarchical clustering with Jaccard and cosine distances underlines the similarity between the two. Conversely, the other values within that same row (column) are lower and much closer to 0.5. The

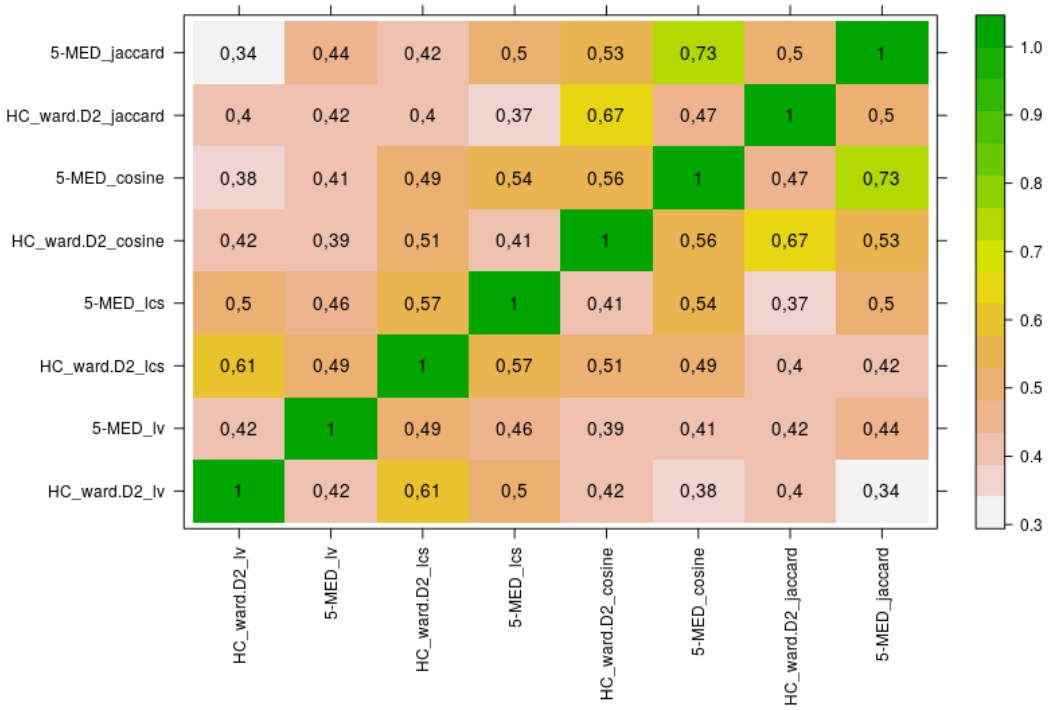


Figure 5.5: F1 score within different HC(hierarchical clustering) and k-medoid approaches with 5 clusters.

lowest value within the row (column) is obtained comparing HC with d_{Jac} and K-medoid with d_{LCS}, with a value of 0.37. Another strong similarity, as expected, is obtained between the two K-medoid classifications with Jaccard and cosine distances underlining their similarity. Other strong similarities are observed within hierarchical clusterings obtained by LCS and Lv. So, in general, distance functions are dissimilar from each other and allow the analysis of the dataset from different perspectives using different clustering techniques. From a technical point of view, results have been obtained with `clusterCrit` package[36] and represented with `lattice` package[119].

Tree field plot With visualization techniques like as tree filed plots, we can gain a quick understanding of how data is related to one another, with a variable number of clusters.

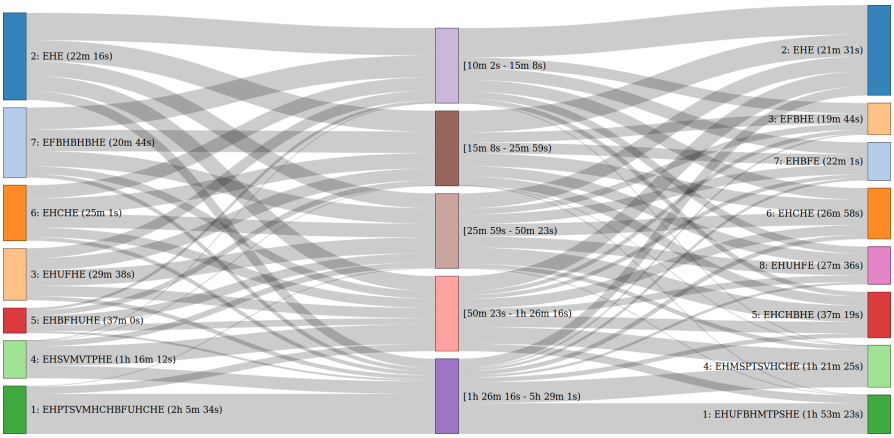


Figure 5.6: Example of a three field plot, for the month of October (time quantiles and medoids).

5.1.2 Castel Nuovo results

Firstly, a description of the clusters is presented, by means of violin plots representing the main numeric features and describing overall the cluster. Further insights on all the numeric features have been given in a second subsection with the correlation matrices and chord diagrams providing relation among all numeric features.

The results are shown in the following figures. As in Fig. 5.7, Fig. 5.8, Fig. 5.9 and 5.10

Cluster description

To analyze the semantics of the clusters and to propose a useful meaning of them, and finally, some considerations have been made by observing the obtained results in Fig. 5.7. Both Figures show how feature values vary within clusters through the box plot within violin plots. The choice of these plots is motivated by the geometry of the distributions of the elements inside each cluster. In fact, it is possible to see multimodal and unimodal distributions, making the representation useful for analyzing the data. To perform the clustering step, we choose the number of clusters as $k = 5$ empirically according to our data and as suggested in [105]. The 5

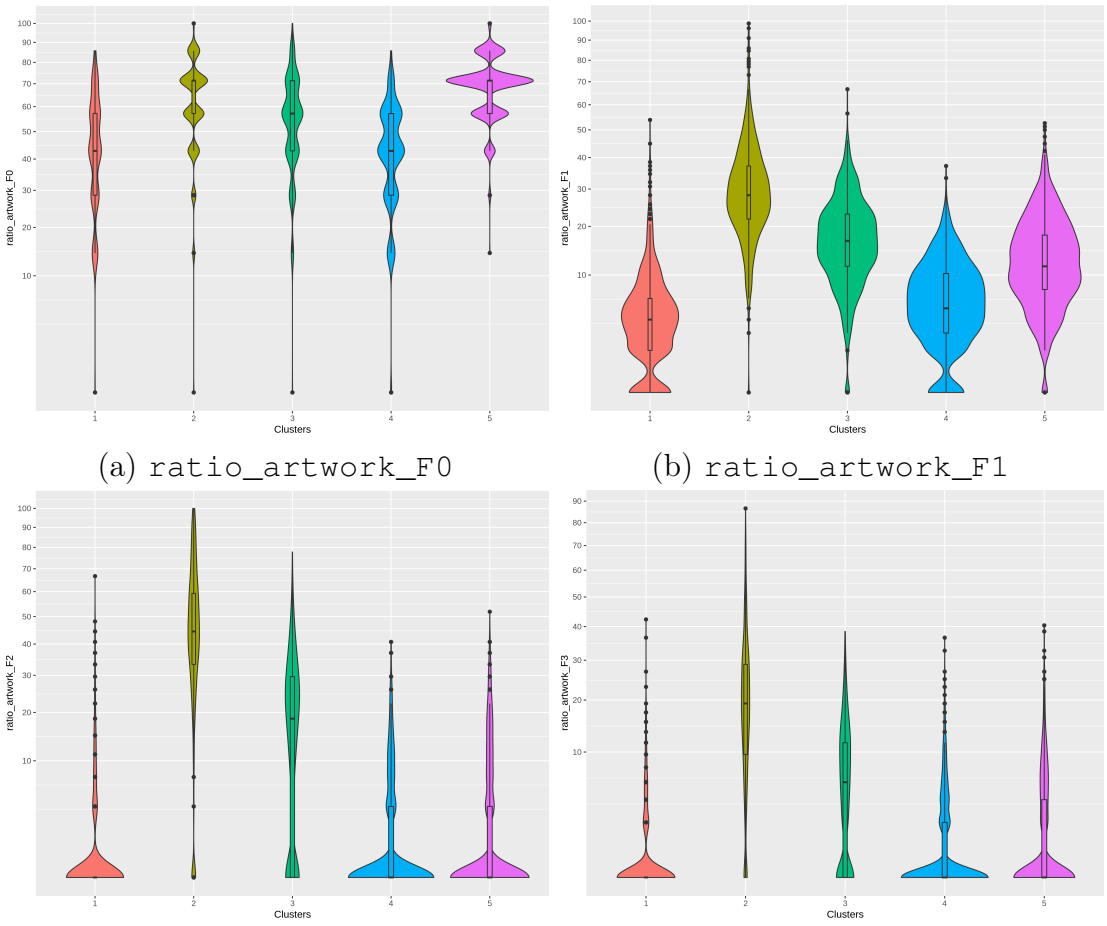
obtained clusters represent different types of visitors, which can be summarized as follows :

1. Stranger visitors (*salmon* color cluster)
2. Strongly engaged visitors (*citrus* color cluster)
3. Moderate visitors (*jade* color cluster)
4. Basic interested visitors (*skyblue* color cluster)
5. Photo-enthusiast visitors (*magenta* color cluster)

To better understand the obtained clusters we deeply analyze Fig.5.7.
Let's consider the distribution of the clusters by the number of the visited artworks categorized by the floor number, illustrated in Fig. 5.7: each floor gives a completely different distribution of the clusters. This relates with Fig. 4.11, where the number of visitors interacting with the beacon is depicted with a heat-map. Overall, more visitors are extremely engaged with the artworks at floor 0 and at the beginning of floor 1, while they reduce their interest as they go upstairs or as they access the *Charles V room*. Clusters at floor 0 cannot be distinguished, as shown in Fig. 5.7a. Moreover, all of them are affected by a large variance in terms of the distribution of the visitors. For the other floors, the narrative is completely different: for instance, clusters 1 and 4 at floor 1, shown in Fig. 5.7b, exhibit very low values, while cluster 2 continues to exhibit the highest values than the other clusters. Interestingly, for the second and the third floor, shown in Fig. 5.7c and 5.7d, there's a net prevalence of people who do not watch any artwork at all, except for cluster 2 and 3. Figures 5.7 suggest a possible classification of the visitors in each cluster, where the cluster are ordered by the size of each cluster.

Correlations among numerical features

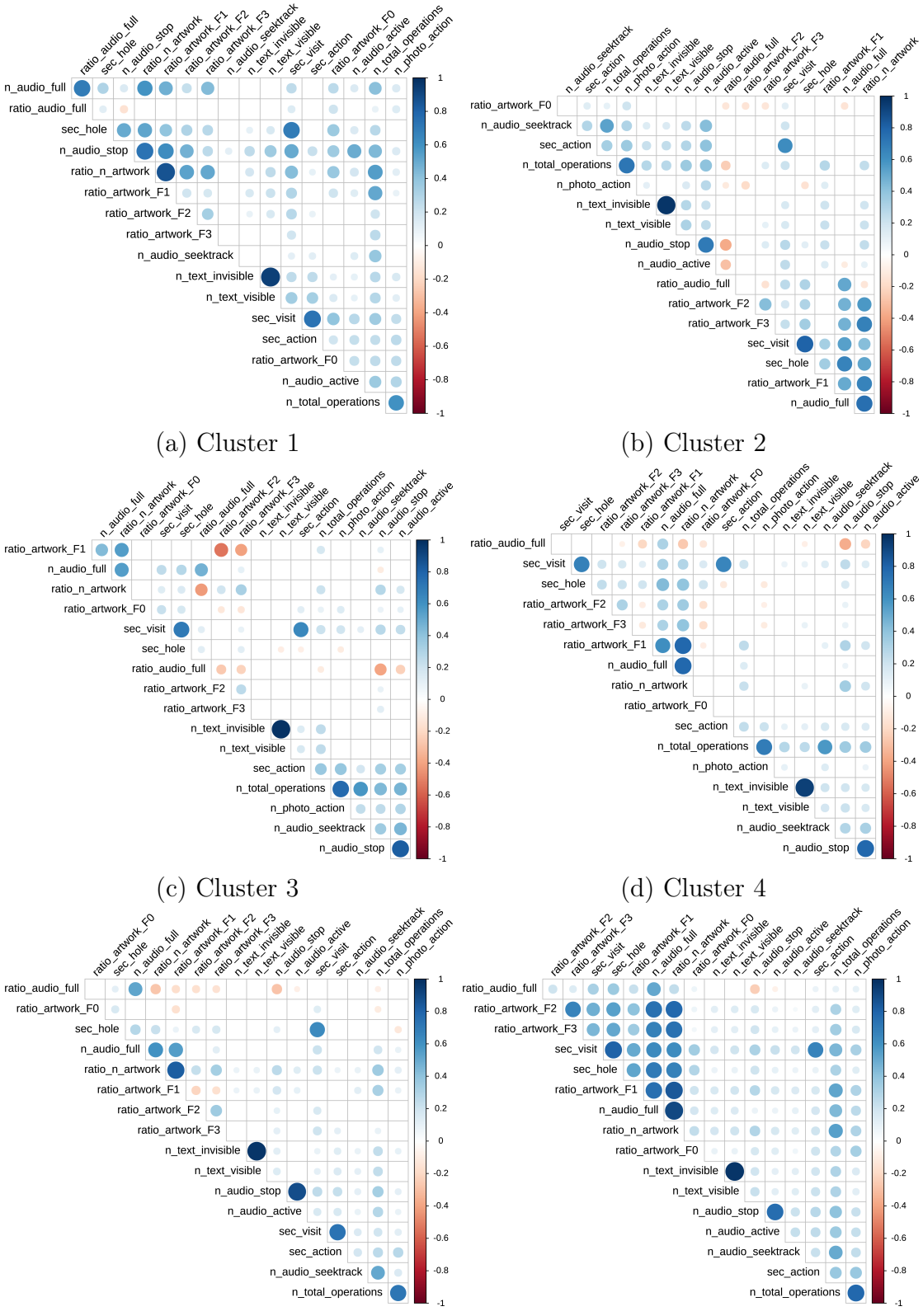
In order to strengthen the quality of the clustering description, correlation matrices were generated, as shown in Fig. 5.8.



(a) `ratio_artwork_F0` (b) `ratio_artwork_F1`

(c) `ratio_artwork_F2` (d) `ratio_artwork_F3`

Figure 5.7: Violin plots of the four variable `ratio_artwork_F*`, i.e. the percentage of the number of the visited artworks in respect to the total number of the artworks for each floor (0, 1, 2, and 3 respectively). The numbers on the y-axis are scaled with a square root scale.



(a) Cluster 1

(b) Cluster 2

(c) Cluster 3

(d) Cluster 4

(e) Cluster 5

(f) Full dataset

Figure 5.8: Grid of correlation matrices on each cluster and for the whole dataset of 4719 elements. Note that the order of the variables in the matrix differs in each cluster, because they are ordered according to a hierarchical clustering of the variables locally to each cluster.

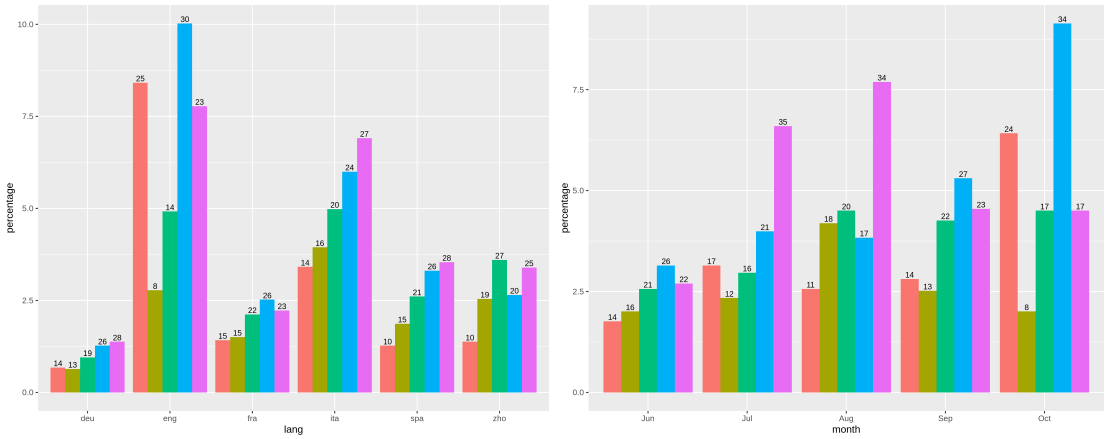
For cluster 1, shown in Fig. 5.8a, there's high correlation between the number of times an audio file is stopped before it ends (`n_audio_stop`) and the number of visited artworks in total (`ratio_n_artwork`). For cluster 2, shown in Fig. 5.8b, the number of actions to the photo (`n_photo_action`) is not correlated to the duration of the visit in the museum (`sec_visit`), which suggests only selective interest to specific photos. For cluster 3, shown in Fig. 5.8c, there's a strong anti-correlation between the number of visited artworks in first floor (`ratio_artwork_F1`) and both the number of visited artworks in second floor (`ratio_artwork_F2`) and the number of visited artworks in third floor (`ratio_artwork_F3`). Moreover, there's no correlation between the duration of the visit inside the museum and the number of visited artworks, which suggests a possibility that visitors in cluster 3 are constrained by time schedule. For cluster 4, shown in Fig. 5.8d, some anti-correlation is present between the number of visited artworks and the number of the listened audio files with full duration (`ratio_audio_full`). For cluster 5, shown in Fig. 5.8e, ignoring the common correlation relationship with the other clusters, nothing results correlated.

Cluster description by categorical features

After a comparison with numerical features, in this section we analyze in Fig. 5.9 and 5.10 how clusters behave within firstly ignored categorical features, i.e. `lang` and `month`.

At the start of the visit, on the welcoming screen, a visitor could choose among 6 languages. Fig. 5.9a illustrates how visitors are distributed through different clusters by the selected language. Visitors who chose English (*eng*) and Italian (*ita*) are dominant with respect to the other languages. Moreover, visitors which chose English can be collocated with high probability either in cluster 1 or in cluster 4, and with much low probability in cluster 2.

Additionally, the visitors which chose Chinese (*zho*) can be collocated with a higher



(a) Bar plot with `lang`. (b) Bar plot with `month`.

Figure 5.9: Bar plots with the categorical features. The numbers on the y-axis, scaled with a square root scale, indicate the number of the visitors related to the feature in ratio with the total visitors. The number on the top of each bar indicate the distribution of the visitors with the feature in each cluster, in percentage.

probability in clusters 2 and 3 than the other visitors. Lastly, the percentage of visitors which are present in cluster 5 is homogeneous with respect to all of the languages.

Fig. 5.9b illustrates how visitors are distributed through different clusters by the month of visit. There is a homogeneous distribution of clusters in June and September. Such distribution does not hold in the other months. Cluster 5 (magenta) is prevalent only in the summer months (July and August). Cluster 2 (citrus) is prevalent only in August. Cluster 1 (salmon) and Cluster 4 (skyblue) are mostly present in October. Cluster 3 (jade) is equally distributed among all the months. To visualize in the same plot both categorical features, a three fields plot[6] is proposed in Fig. 5.10. The three fields plot is available in `R` and is based on the *Sankey diagram*. In Fig. 5.10 the flow can be read from the center to the left and right columns.Behavioral aspects can be inferred by observing this kind of plot, e.g. considering the cluster 5 coherently with our expectations, visitors most present in August, are equally spread through all languages.

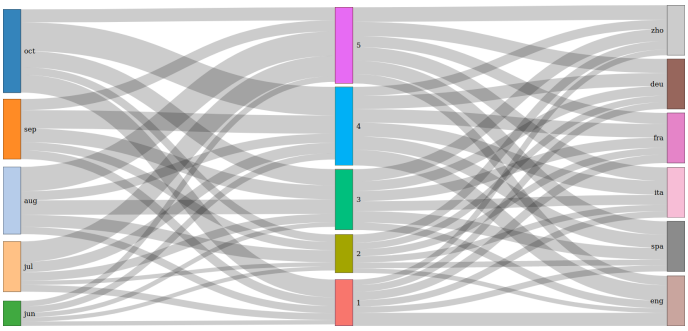


Figure 5.10: Three fields plot, relating `month` and `lang`. Each cluster is represented in the center column, accordingly with previously used colors. A cluster is linked to both the month and language, shown on the left and right column, respectively.

5.1.3 Comparison

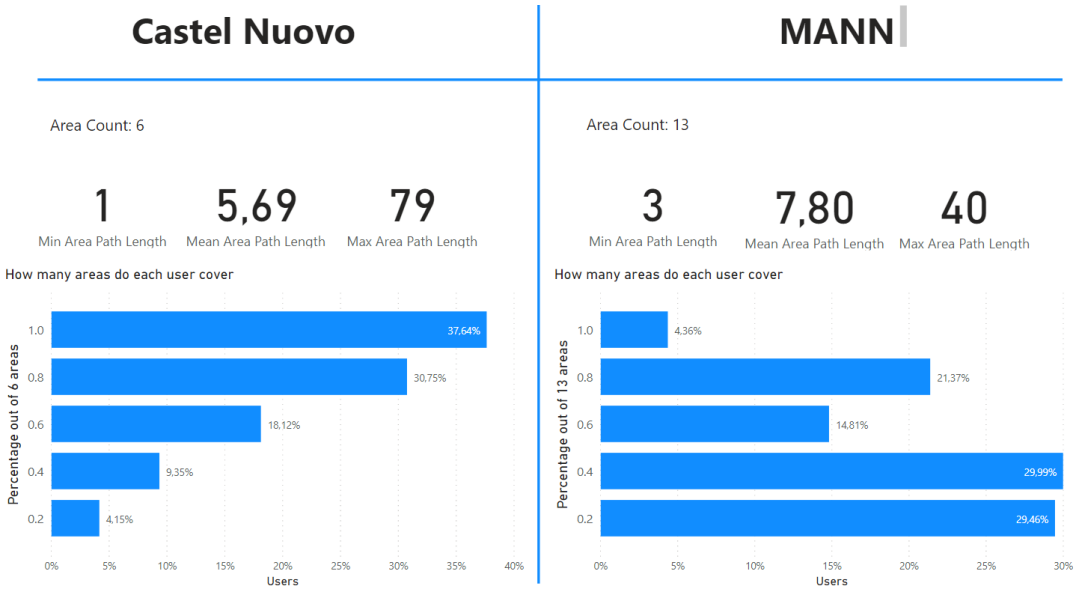


Figure 5.11: PowerBI statistical comparison of Castel Nuovo and MANN.

Statistics

Knowledge Graph Model Using a KG to organize data can be an effective way to quickly, easily, and directly discover information. The first step is to model the schema in a fourth normal form that will allow it to be queried against a NoSQL database and presented as the first layer of results. A subsequent step will

be to apply knowledge graph embedding, which arranges the entities in the KG in a vector space, followed by machine learning techniques.

A KG graph approach is presented for the purpose of assisting museum stakeholders with services by extracting knowledge and new insights. The KG is a unified model for both MANN and Castel Nuovo data. The model included parameters for the distances between a museum's Points Of Interest, which were calculated using R, and statistical data about user behavior was presented using PowerBi. (Fig. 5.11) The end result is the identification of aggregate behaviors.

Technically, this framework is based on GRAKN, a novel and intelligent graph database capable of modeling large datasets, and Amplighrph, a graph embedding system capable of multiple embeddings. A similar procedure was performed on booking data from the Naples Local Health Department and presented at the Grakn Conference. This can extend the embedding result to Cultural Heritage data: spatially model the museum's nodes for group identification within the museum cluster of typical days.

Typically, Data Science processes require organizing and cleaning the data, as well as structuring the data (shown in blue) in a way that the corresponding algorithm can use it (in green in the Figure).

The methodology to explore is to remodel the data as a Knowledge Graph and then apply graph techniques such as node importance and community detection. In the model implemented in Grakn, where entities are represented by ovals. The node that collects the information, along with its location attributes, and which may also contain a semantic description of the artwork contained within. And the path taken by the visitor, th that abstracts all possible paths taken by visitors, as well as the overall route taken by a device. Notably, in contrast to standard KG, here we are not referring to model text, but to the relationships between columns in typical databases.

What makes this approach interesting is that it would make it simpler to

model n-ary relationships by grouping relevant pieces of information that express higher-order information.

Thus, rather than performing traditional topology-based network analysis with adjacency matrices, the application of embedding techniques can actually vary the final output. Although both techniques have the potential to produce the same types of applications: Classification of nodes, Detection of community, Prediction of hyperlinks, The significance of nodes, Distance between networks Evolution of networks.

The similarity of nodes' embeddings indicates their network similarity. It appears to me that by leveraging embedding, I could easily apply all that is known about machine learning to real-world data. Thus, the purpose of embedding is to reduce the dimension of a network by mapping each node to a low-dimensional space. Typically, there is a score function.

There are numerous Knowledge Graph Embedding algorithms in the literature, and selecting the most appropriate one for a given dataset is a significant challenge. The TransE embedding is one of the simplest. This is a translation algorithm. It is predicated on the notion that in a triple, one entity is moved by another entity's relation to it. As a result, this algorithm optimizes the alignment of all entities according to their relationships. As a result, it is referred to as a Translational embedding algorithm. Bilinear embedding is another type of embedding algorithm. It is based on the tensor space decomposition. CompleEx, DistMult, and RESCAL are a few examples. Obtaining a high-quality model is not always straightforward. Not to mention that the Knowledge Graph Embedding algorithm requires configuration of additional input parameters.

Most importantly, we did not discover an overall view - analysis of the embedding that described how well it fits various types of data and the relationships between those data and the Graph structure, as well as the mathematical properties preserved by the embedding.

And this is what this table illustrates. As a preliminary findings, there are some machine learning experiments on embedded values, that may be a generalizable result. One can observe distinct partitioning, and the primary output is a relationship-based grouping.

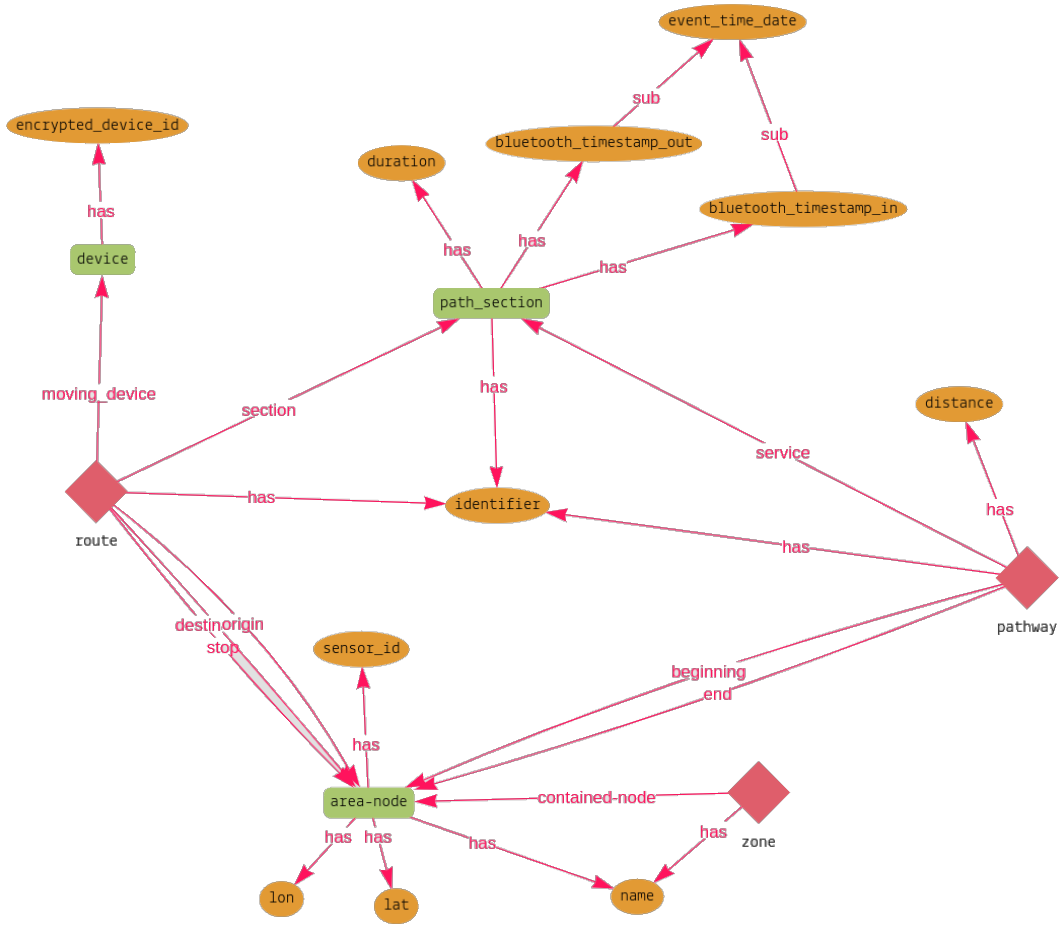


Figure 5.12: Knowledge Graph modeling visitors paths in a museum.

It would be advisable to looking of a mixture of embedding models, like ConvE and RotatE). Another possibility is to completely rewrite the code and use DeepMind's Graph Nets. Additionally, there are possible investigations into the application of reasoning to the KG. And there is still a great deal of work to be done on link analysis and link prediction.

5.2 Healthcare administrative data

This section presents the application of the first two steps of the proposed framework in Fig. 4.13, from the tabular data to the KG. The data underlying such a process include descriptive statistics and analyses of the data quality, like identifying missing or invalid data. Considering that our case study originated from a real-world scenario, namely a dataset relating to the booking of medical appointments prescribed by practitioners, the data came from legacy databases and web services. This fact raised many issues that we will describe in this section. Working with the data, and also with business experts, gave us a good understanding of such data, enabling us to generate a KG.

5.2.1 Data description

Referrals and medical services within the public healthcare system in Italy are accessed through a Booking Centre administered by the local health authority and controlled by the regional government. The data that we have analyzed in our research study came from a distributed database relating to different local health departments of the Campania region of Italy. In more detail, we relied on data generated from the last six years of medical prescriptions, and the booking of appointments, including cancellations and reschedulings, which amount to more than 20 million entries.

Such data, coming from a legacy SQL-like database, have been usually used as a bunch of data, without any entity-relation schema. The standard booking process uses essentially three tables: a booking appointment data table, a cancelled booking data table, and a paid appointment table. One of the main issues that may be encountered is the lack of an entity-relation structure. The data are stored simply as a long sequence of rows, usually repeating many values in similar columns and rows, including possible errors.

The main table has 33 attributes, indicating the name, gender and age of the patient, and the identity of the health service where the booking was made, sometimes reporting the name of the practitioner who made the referral and the specific date of the appointment. In this kind of table, it is also possible to find two almost identical rows, which may differ in only one or two columns. The second table, also contains 33 attributes, sharing the meaning of 31 with the booking appointment table (the main table), with merely one column changed in meaning from booked to cancel. This means that this database is essentially a subset of the first table. Finally, the third table contains 38 attributes, sharing 25 of them with the booking table, with others being added to specify the payment transactions. In our research study, we focused our work on the booking appointment table. The subset of columns that we selected is reported and described in Table 5.1.
The other two datasets, containing instances of the pairs, `medical-branch-id` with `branch-description` and `health-service-id` with `health-service-description`, were also also provided for descriptions support.

The database structure 6 years of medical prescriptions, appointment booking, cancellation and rescheduling totals more than 10 million entries. Main table: 33 attributes identifying which patient, whose gender, which age, which health service was scheduled, which practitioner referred the patient, and which booking operator booked the patient for a given day. How was the data on the database saved? Regrettably, all of the process data was recorded as a series of columns. We could locate things like practitioner id, date of referral, and date of booking because we had primarily a single table with all the attributes. The data was not structured as an entity-relationship database, but was simply used as a collection of rows and columns with no logical control. As a result, we had to first comprehend all of the discrepancies and then cleanse.

Field-Name	Type	Explanation
booked-date	date	date of the booked appointment
reservation-date	date	date of the first contact with the booking staff, and also the date of the insertion of the record in the database
last-reservation-change-date	date	last update of the reservation, which usually matches the reservation-date
encrypted-nin-id	string	encrypted national insurance number, used as a non-decriptable numeric ID of the patient, with a value of -1 representing an undefined patient
gender	long	1 for male, 2 for female, -1 for undefined (i.e. for anonymous patients)
patient-age	long	age of the patient at the time of the issue of the referral, with a special value of -1 for anonymous patients
nuts-istat-code	long	Italian nuts code, identifying the local administrative areas, possibly linked to the postal code
booking-agent-id	long	anonymized numeric ID of the booking agent
medical-branch-id	string	alphanumeric string that contains a code of the medical branch
health-service-id	string	alphanumeric ID of the health service
practitioner-id	long	anonymized numerical identifier of the prescriber, with a special value of -1 for undefined practitioners
referral-centre-id	string	identification of the unit that delivers the appointment, being a concatenation of the *dispenser* and the *surgery*, separated by a hyphen
appointment-encoding	string	concatenation of the local-health-department-id and the appointment codification
booking-type	string	numeric value of 1 is for direct check-in (e.g. from a hospital during treatment), 0 is for the conventional booking reservation
referral-id	long	anonymized numeric ID of the referral
referral-date	date	the date of the issue of the referral from the practitioner to the patient
priority-code	string	the code of the priority of the referral
exemption-code	string	the code of any exemption with respect to the patient and the health service
number-of-health-service	long	number of necessary health-service-ids, rarely greater than 1
local-health-department-id	long	sub-regional local health department. In our data set there are three departments, identified with letters A, B and C

Table 5.1: Data set features. The table contains descriptions of the columns of the local health department booking centre.

5.2.2 Data issues

This heterogeneous kind of data presents a significant degree of inconsistency, given the non-entity-relation structure. In this section, we report some of the main issues resulting from this inconsistency.

First, the branch table had repetitions of codes with the same description or very similar descriptions. For example, considering the Cardiology branch, it was described by three different codes, 02, 002, and CAR with also some differences in the description in the sense that it was sometimes written in upper case and sometimes in lower case. This problem has been solved by sanitizing and standardizing the string descriptions. The reason for this mistake is that, initially, the local health department office modelled the branches of the glossary by inserting codes formulated at the insertion phase, during the transaction from the paper to the computer system. Subsequently, the codification changed to numerical values, and finally, a few years ago, with the introduction of a regional catalogue, in addition to changing all the health care services codes, also branch descriptions were standardized, re-organized or merged.

In the health service table, there were codes and descriptions not found in any standard code from the official list of the Campania region. They could be divided into two groups. Some of them were referable to the official codes through manipulation of the identifier or the description (e.g. replacing dashes with dots, or eliminating content within brackets in the description). Others, which occurred thousands of times, cannot be traced back to any service offered by the region, for example, a *to be cancelled* description or *home visits*. All these cases, grouped together, amounted to *two million* distinct lines in the dataset. This can be explained by the fact that also for health services, the glossary has evolved over the years, and therefore the codes have also changed. Moreover, to simplify their work-life, booking staff managers have sometimes changed and simplified the performance descriptions. Furthermore, the local health department operates under a National Health Service

contract using the official regional catalogue. However, at the same time, the local health departments have continued to accept old codes for other contracts that were not related to the national contract, in order not to change the configurations of the thousands of services offered. They have preferred to continue to use the old codes. It means that an old code, intended only for non-NHS reservations, may be present in the database, for all reservations, up to a certain point of time.
We also noted that some referral identifiers were connecting more than one patient, or more than one practitioner, ids. It is impossible since it is expected that a referral is always associated with a single doctor who issues it for a specific patient. What we eventually realized is that the dataset was also used for testing by the managers of the health service department. When a referral did not have a valid appointment, it could be reloaded, and in particular, made by the operator who inserted the patient and the doctor manually. Therefore, in the tests, the same code was used for various patients and various doctors.
In summary, the overall impression of this database, coming from the real world, that we analyzed and studied was that it was a set of rows and columns, used by operators, as an old ledger, without any specific consistency controls and verification. This was due to the fact that the same database was used for many years while there were at the same time many nomenclature changes. The regional and national institutions have regulations and codified indexes for health services which have changed over the years, just as the taxonomies have changed. Moreover, some codes are nationally adopted, and some are region-specific. All these problems were overcome by a cleaning process, and a semantic extraction of the data, thanks to a data analysis phase and the support of domain knowledge experts.

Text cleansing In terms of the data cleansing process, we had to keep in mind that our tabular data contained indexes and descriptions that, while theoretically standardized, were sometimes manually updated by human operators. We added a flag column to examine what Grakn would see if he only looked at the connection in

the complete graph in future analyses. We were able to maintain the previous fomr and the cleaned foms, which were put together in different insances, so that we could later examine how we could match them in a graph by fagging. In terms of the data cleaning process, it was necessary to consider that our tabular data had indexes and descriptions that, while theoretically standardized, were sometimes manually updated by human operators. Spelling problems, name convention alterations, and text contractions added by operators were also discovered in the dataset. To address this issue, the string descriptions were standardized using a Python script and a string matching method. We created and developed a Python process that used string matching to group comparable strings whose variations could have been caused by spelling mistakes, name convention changes, or text contraction. Finally, we added flag-columns indicating the quality of each database row, which would be used in later analysis. There were codes and descriptions in the health service table that were not located in any standard code from the Campania Region's official list. They could be classified into two categories. Some of these can be linked to official codes by modifying the identification or description (e.g. replacing dashes with dots, or eliminating content within brackets in the description). Others, which occur hundreds of times, cannot be linked to any service provided by the Region, such as a canceled description or home visits. When all of these cases were added together, they amounted to nearly two million distinct lines in the dataset.

Cross-field validation The data were tabular and not organised into an entity-relation database; an in-depth investigation of the relationships between pairs of rows with the identical values on one or more fields was done. Given the non-entity-relation structure, this heterogeneous type of data contains a great deal of inconsistency. First, consider a circumstance in which the same identical record was sent to three distinct medical disciplines. Another example would be the same referral having two separate change data. The same referral id, two different practitioners, and patients. We also discovered several referral identifiers

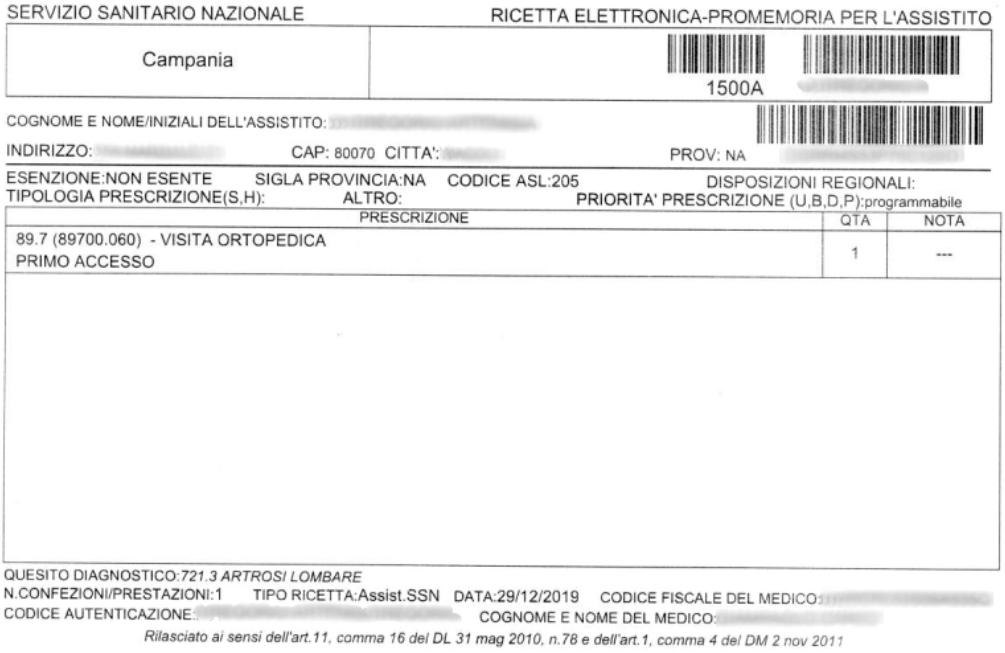
that linked more than two patients or more than two practitioner ids. It is an impossible circumstance since a referral is always associated with a single doctor who gives it for a specific patient. What we subsequently discovered was that the dataset was also used for testing by the health service department's supervisors.

5.2.3 From raw data to a Knowledge Graph schema

Within the dataset, also spelling errors, naming convention changes, or text contraction added by operators were found. To overcome this issue, a standardization of the string descriptions was performed through a Python script with a string matching algorithm. This process was applied to a couple of attributes, health service id and description. After measuring the confidence of these matches, we took only the ones with good confidence and labelled the rest with a flag. The same approach was used for the medical branch attribute. Additionally, we had to fix a branch named *other*, which was also used by the booking staff to categorize services not belonging to any other available branch. The use of flags, which indicate the quality of the data, can be useful for the link prediction and identification in further studies and investigations.

After the cleaning process, other attributes were created: (i) a `refined-health-service-id` representing a correction of the `health-service-id` through a merging of the strings referring to the same health service and flagging unknown labels with a placeholder. A similar approach was used with a `refined-medical-branch-id`, enhancing the identification of the `medical-branch-id`, and with the `official-branch-description`, which is a standardization of the words in the `branch-description`. Finally, since the relationship of the referral is the core of this work, we added, in a similar way, a `referral-modified-id` adding more knowledge to the `referral-id`, by grouping all the possible test referrals and all unknown and unpolished data into a placeholder.

Now, a question arises, namely, how to analyze such data and extract useful insights? We have worked in close contact with doctors, medical staff and the IT support system to better draw inferences from the available data. Thanks to the doctors, we were able to understand what kind of relations existed between the patient, the practitioner and the referral. An overview of a classic referral is shown in Figure 5.13 In relation to the main concept of the entity-relation, a


SERVIZIO SANITARIO NAZIONALE

RICETTA ELETTRONICA-PROMEMORIA PER L'ASSISTITO

Campania

1500A

COGNOME E NOME/INIZIALI DELL'ASSISTITO:

INDIRIZZO: CAP: 80070 CITTA': PROV: NA

ESENZIONE:NON ESENTE SIGLA PROVINCIA:NA CODICE ASL:205 DISPOSIZIONI REGIONALI:

TIPOLOGIA PRESCRIZIONE(S,H): ALTRO: PRIORITA' PRESCRIZIONE (U,B,D,P):programmabile

PRESCRIZIONE	QTA	NOTA
89.7 (89700.060) - VISITA ORTOPEDICA PRIMO ACCESSO	1	---

QUESITO DIAGNOSTICO:*721.3 ARTROSI LOMBARE*

N.CONFEZIONI/PRESTAZIONI:1 TIPO RICETTA:Assist.SSN DATA:29/12/2019 CODICE FISCALE DEL MEDICO:

CODICE AUTENTICAZIONE: COGNOME E NOME DEL MEDICO:

Rilasciato ai sensi dell'art.11, comma 16 del DL 31 mag 2010, n.78 e dell'art.1, comma 4 del DM 2 nov 2011

Figure 5.13: Referral example from the Italian public healthcare service. In the image, all the attributes of the referral can be noted, as well as the attributes of the entities related to it. Starting from the top, the first block indicates the region, in this case, Campania, and the bar codes identify the referral. The second block of data shows the name, address and city of the patient. It reports, in order, the exemption code, if provided, the code of the local health department and its area, and the booking type and its priority code. The referral reports in the centre a list of the prescribed health services (*la prescrizione*), with the number of needed services (usually 1). Finally, the last block of data at the bottom, reports the overall nature of the prescription, with the motivation of the diagnosis or symptom, and the date when the referral is issued and the practitioner data.

KG builder could consider Figure 5.13. This referral is a relation that relates: (i) one practitioner to one patient, and (ii) the referral itself to n health services prescriptions, and as a consequence, (iii) to a specific medical branch.
Usually, in a KG, relations are seen as links among nodes. What Grakn introduces is the possibility of easily modelling a schema with hyper-relations. Each hyper-relation can have its own attributes, and each hyper-relation can also relate two or more entities. Each entity, with attributes, can play one or more roles in one or more hyper-relation. For the sake of simplicity, we will generally refer a hyper-relation as relation, since we will not be using the idea that a link among nodes is a relation but, rather, that it *has* a predicate.
In Grakn we obtained the schema that depicted the referral relations between the practitioner and patient. On top of this relation we constructed the *second-level* relation, the reservation, that links the referral with the health service provision, the appointment provider and the booking agents. In Figure 5.14 the KG schema is shown.
Our technique led us to write a Python code, that semi-automatized the KG construction. In detail, the script automatized the dump from the SQL tables to the Cassandra DB thanks to Grakn's Python API. Besides, we created batches by using the *Graql* query language to match and insert new data. The dumping populated all the attributes and two relations, the referral and the reservation. After this step, a *reasoning* process was applied to derive the `provision` and `health-care` relations. These two relations were derived by using the rules shown in Figure 5.15, expressed in the Grakn language named *Graql*.

Given a KG populated with its data, the process of gathering insights can start. First, an embedding phase has to be performed in order to obtain real value data the ML algorithms. The main reason for the embedding step of Fig. 4.13 is to obtain the real value data. Embedding is the best-known technique to pass from a graph and express its nodes and links in a numerical space, usually $\mathbb{R}^K$. An

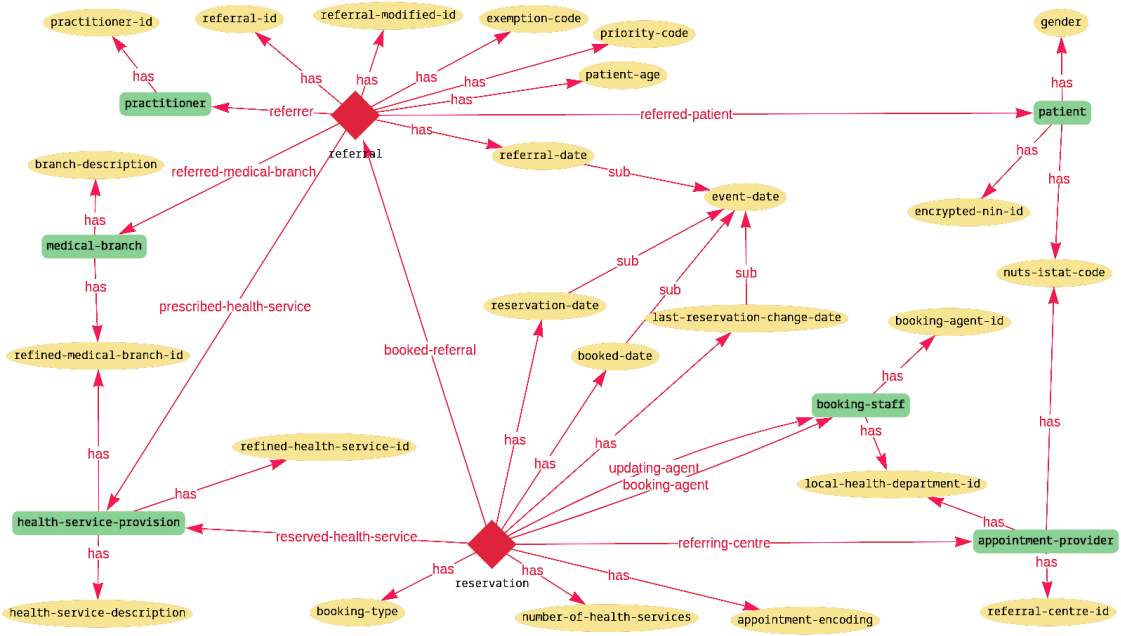


Figure 5.14: Knowledge graph schema. Ovals represent attributes, rectangles entities and diamonds relations. Each relation relates to least two entities, and such entities play some role in the relation, which is specified on the arrow connecting the entity-relation. The referral relation relates a patient and a practitioner, trough the `referred-patient` and `referrer` role links. Because of privacy purposes, there is no date of birth of the patient in the database, but its age at the time of the issued referral, so the relation, not only has a referral date but also the age of the patient at that time. The second relation, reservation, add to the schema to a nested-relation, since it relates no only entities but also the referral relation, that plays the role of being a `booked-referral`.

```
## Relation rule: provision ##
referral-centre-provides-health-service sub rule,
when {
    (reserved-health-service: $hs, referring-centre: $rc) isa reservation;
}, then {
    (health-service-provider: $rc, provided-health-service:$hs) isa provision;
};

## Relation rule: health-care ##
patient-is-cured-at-health-care-provider sub rule,
when {
    (booked-referral: $ref, referring-centre: $hcp) isa reservation, has booked-date $bdate;
    $ref (referred-patient: $p) isa referral;
}, then {
    (cure-provider: $hcp, cured-patient:$p) isa health-care;
};

patient-is-cured-at-health-care-provider-at-booked-date sub rule,
when {
    (booked-referral: $ref, referring-centre: $hcp) isa reservation, has booked-date $bdate;
    $ref (referred-patient: $p) isa referral;
    $hc (cure-provider: $hcp, cured-patient:$p) isa health-care;
}, then {
    $hc has booked-date $bdate;
```

Figure 5.15: The rules added to the KG. The first, generates the `provision` relation, when a health service is reserved and booked by a health service provider. This new `provision` relation is at the same level as the one from which it is generated, i.e. `reservation`.
The `health-care` relation moves across the two relations, `reservation` and `referral`, assessing where a patient receives treatment. These rules add two more relations (represented by diamonds), not shown, in Figure 5.14.

extensive literature review on embedding techniques can be found in [20;123] Within embedding, *similar* nodes are mapped in close points in the space. The way in which this process is performed defines the specific algorithm. We relied on a graph embedding like AmpliGraph [31], GraphVite [158], and Pykg2vec [152]. Being to our opinion AmpliGraph the most stable and updated solution at the time of the writing. In our framework, we have selected Amplighrap's ComplEx embedding [31;137] due to its quality, updates, support and open-source code availability. ComplEx is based on *complex* embeddings that can handle a *large* variety of binary relations. It is *scalable* to large datasets as it remains linear in both space and time. Furthermore, it is *simpler* with respect to other alternatives, as it only uses the Hermitian dot product. Given a fact $r(s, o)$, a relation r between a subject s and an object o, their respective embeddings e_s and e_o belong to $\mathbb{C}^K$, where K is the desired embedding space dimension. Other embedding algorithms to be mentioned include TransE and TransR [54]. The authors in [1;72] propose CETransR, which extends TransR first by modelling the entities and relations in distinct vector spaces and then by clustering the same knowledge mentions into groups.

Schema generation: Entity definition From this scrambled database, it is necessary to understand what each clause means, and not always what is written in a document related to a database, what is on the database, and how it is actually utilized. As a result, we required a method to comprehend the database's hidden relationships. Rows in a table that have the same value on a fixed field may be linked to the same relation or entity. Entity disambiguation was accomplished by counting the unique occurrences of pairs of columns in the database in order to gain a general understanding of which pair of fields may be deemed belonging to or not belonging to an entity or relation. This procedure was generalized to all columns; here is an example. In this example, we have a notion that the patient's age can be an attribute. What happened with our data is that we did not have the patient's year of birth, but the age when he booked

the health service, therefore the age had to be the main attribute of the referral. We have a table with the patient's id, the gender's id, and the practitioner's id. We count the number of unique occurrences for each row in the database. Then we examine the unusual occurrences for couples. These must be completed for all columns, and the differences from the previous step must be examined to determine when this number does not change or is more or less comparable. Not everything could be done in the most "data-driven" way feasible. There are numerous studies that attempt to do this.Finally, this method was modified with the help of several domain experts to produce the final knowledge graph schema.

Schema generation: relation understanding We worked closely with doctors, medical staff, and the IT support system to gain a deeper understanding of the available data. So we had to collaborate with practitioners and look at typical data processing in the healthcare system to extract correct information, or what an operator would truly desire, rather than what the database was created for the finest data we had were prescription references. We were able to comprehend the nature of the relationships between the patient, the practitioner, and the recommendation thanks to the doctors.

Here we look at an example of a patient and a practitioner in a referral relationship in a Grakn-like approach. Because of the multidimensionality of the data linkages, we can see why Knowledge Graphs were useful. Most documents you may use are intrinsically a meta-relation (in the Grakn sense), therefore we could envision taking such a document and imagining it as an augmented reality (AR) example, where we take ordinary paperwork and argue it with relation-entity-attributes.

The practitioner's id, and this is about the service and branch of health service. It should be noted that a single health service might be assigned to more than one branch. The second connection is regarding booking. Now we join the entities using the referral relation and the referral's characteristics, such as the exemption

code and id/Note that the patient age is an attribute of the referall, not the patient. We didn't have the patient's birth year, but we did have the age when the referall was issued.The booking is then sent on to the agent who books the health service, as well as the health service provider, where the patient will have his blood test, operation, and soon Finally, there is a second level meta Grakn meta connection, that relates, booking staff, appointment taker, and the previously blue referral, as well as the health service You could book various health services in a referral at various times or with several providers.

More rule-based meta-relationships have been added. The rule addition simplified database processing because all entities are everything on the database, but the king of n-1 n-n relations were multiples. This is the final schema schema that we created; it is not a made-up schema, but rather a real-world solution to a real-world problem.

5.3 Results with GRAKN

Workstation Hardware Since we began with 10M entries, we had to do all of our work on a Workstation that met the following requirements. Most of the timings and sizes we display are connected to this hardware, namely a 16 core Intel CPU @ 3.60 GHz. For multidimensional cleaning and machine learning, the RAM size needs to be increased. A high-performance video card is required for embedding, especially as the number of triples grows.

Data Storing Once the data has been categorized, cleansed, and a semantic structure developed, it must be transported to the pipeline's second step: data storage in a graph database. Each record in a relational database is effectively an entity. A relational database, in reality, has a ledger-structure. When we employ foreign keys, we hope to establish relationships between those qualities. In particular, several json files were made.

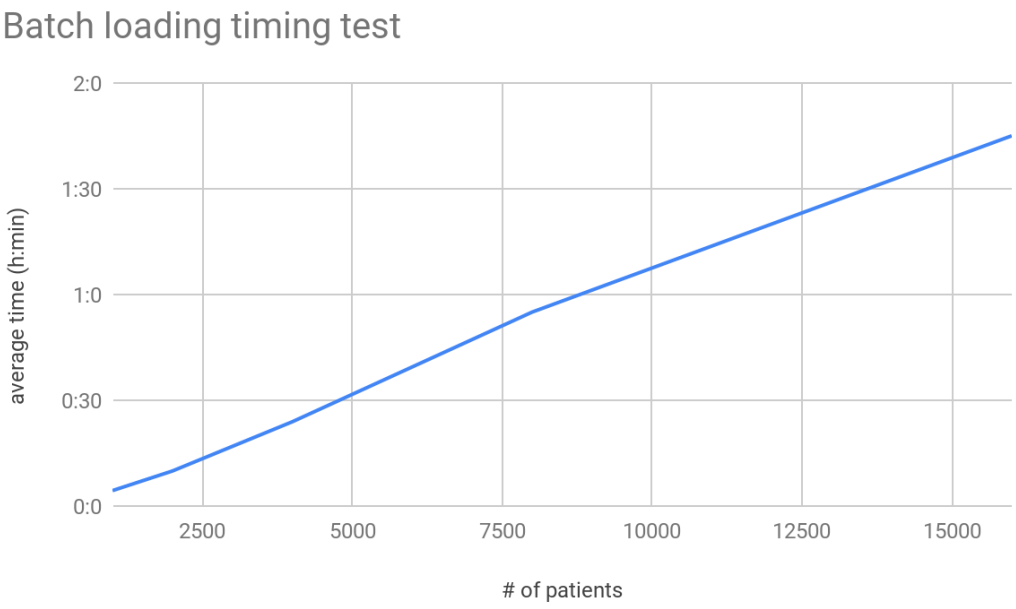


Figure 5.16: Loading Time. Specify problem relates no to query. But to complex matches prior insertions.

5.3.1 Scalability test

We discovered that the insert time grows linearly and that the relation insertions time is usually constant, but only for a limited number of insertions.

Here are some instances of results, including execution time (all of them are done in parallel with 8 processes): 2k entities, 7k relationships: total time 4min 3000 entities, 15000 relationships: Total time: 10 minutes 5k entities, 35k relationships: total time of 30 minutes However, when we attempted to insert 350k relations, the execution was so long that it was interrupted at some point (3 days later), and some relations took 200 seconds to enter. Some testing was done with 90k relations, and a preliminary estimate of the execution time implies that it may take more than 12 hours, and the variability of the insertion time grows over time, and on average, certain searches are rapid (from 0.1 to 2 seconds), while others can be quite slow (more than 9 seconds).

We have a few theories as to what was going wrong: Insertion through the Python API may be fundamentally sluggish. It is unknown whether version 1.6.0

resolves the issue, however the upgrading is on hold for the time being because we do not have much time to work on the embedding component. During our Python API development, if we attempted to create a relation that did not match any object inserted previously, it would simply skip it with no reporting on the Python prompt. The ComplEx method with a dimension count of 300 produced the best results for us.

5.3.2 KG insights

In this section, we present and discuss the results in terms of the insights retrieved through the unsupervised and supervised learning techniques. In particular, the main goal has been to exploit the Knowledge Graph Embedding (KGE) to extract new and useful knowledge from the data. Indeed, the objective has been to take advantage of the KGE since it represents entities and relations as embedding vectors in a semantic space also containing semantic information that can be exploited in ML applications. As a technical detail, all the experiments were performed on a Core i9-9900K - 128GB DDR4 equipped with a Geforce RTX 2080 Ti graphic card. Although we were using a high-end machine, it was not feasible to process the entire available dataset. We started testing the proposed framework by selecting randomly 100 patients, who generated 2k entities and 7k relations, and gradually scaled up. The main bottleneck was the time necessary to dump the database into the Grakn database, version 1.5.9, since the embedding algorithms took several hours to complete the epochs, as we were testing different hyper-parameters. We eventually realized a problem in the query planner, whose performance degrades as the number of links and nodes grow. Developers are aware of such randomness in their performance and are working on improving those issues. Moreover, the video card memory usage of the KGE task started to run out as the triples were growing. As the number of patients increased, the number of triples increased also, and this made the embedder stop working as it ran out of memory on the

11GB video card memory. Obviously, at the same time, we were looking for the highest number of patients to analyze so as to improve the ML algorithms and the related insights. In the end, we extracted $1,747$ patients randomly from the dataset. Each patient was requesting a minimum of 60 health services, but no more than 100. Noteworthy basic information in relation to our final dataset concerns the number of triples, namely $1,558,700$, which depends on to the number of the overall instances reported in Tables 5.2, 5.3 and 5.4.

5.3.3 Knowledge Graph Embedding

In the literature there exist many KGE algorithms. Therefore, selecting the most appropriate one for a given dataset is an important challenge. In this perspective, the authors in[5] discussed four different algorithms, namely TransE, TransR, TransD and ComplEx, and selected TransE to provide some recommendations. To select the most suitable KGE algorithm for our dataset, we analyzed and followed the previously cited approach. It is not always straightforward to obtain a high-quality model, with sufficient data to draw useful and suitable insights. Each KGE algorithm has different input parameters to be set up. In this perspective, we applied a random grid search, which represents a suitable choice, as stated in[12]. We tried multiple train-test-validation sizes, ranging from a 60-20-20 to an 85-10-5. We relied on an Adam optimizer algorithm; the batches varying between $[32, 64, 128]$, the epochs $[100, 200, 300]$, the k $[100, 150, 200]$, the η $[5, 10, 15, 20]$. Where the η (`eta`) is the number of negative, or false triples that must be generated at training runtime for each positive, or true triple. We also tested three kinds of loss functions: the pairwise, the negative log-likelihood loss and the multi-class NLL Loss with the margin selected among $[0.5, 1, 2]$. Considering the Adam optimizer algorithm, we used as regularized Lp1 and Lp3, with $\lambda = 10^n$, where $n = 1, 2, 3, 4, 5$ and a learning rate $= 10^n$ or $= 0.5 * 10^n$. The best results that we obtained are related to two KGE algorithms, as shown in Table 5.5.

@has-relations	Number of instances
@has-booked-date	149,198
@has-reservation-date	122,744
@has-number-of-health-services	122,744
@has-appointment-encoding	122,744
@has-booking-type	122,744
@has-referral-id	31,679
@has-referral-modified-id	31,679
@has-patient-age	31,679
@has-referral-date	30,630
@has-exemption-code	21,545
@has-priority-code	10,958
@has-nuts-istat-code	3,584
@has-local-health-department-id	2,907
@has-practitioner-id	1,991
@has-referral-centre-id	1,837
@has-encrypted-nin-id	1,747
@has-gender	1,747
@has-refined-medical-branch-id	1,337
@has-health-service-id	1,316
@has-booking-agent-id	1,070
@has-health-service-description	1,023
@has-refined-health-service-id	1,023
@has-official-branch-description	131
@has-medical-branch-id	131
@has-branch-description	131

Table 5.2: Number of instances in the @has type relations, these being the ones that express a property of either an entity or a relation (i.e. a hyper-relation).

Relation-entity links	Number of instances
reserved-health-service	122,744
updating-agent	122,744
booked-referral	122,744
booking-agent	122,744
referring-centre	84,673
referrer	31,679
prescribed-health-service	31,679
referred-medical-branch	31,679
referred-patient	31,679
cure-provider	9,537
cured-patient	9,537
provided-health-service	9,471
health-service-provider	9,471

Table 5.3: Number of instances of relation links.

Entities and meta-relations	Number of instances
reservation	122,744
referral	31,679
health-care	9,537
provision	9,471
practitioner	1,991
appointment-provider	1,837
patient	1,747
booking-staff	1,070
medical-branch	1,023

Table 5.4: Number of instances of entity kind.

Model	**MR**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
TransE	26,922.82	0.03	0.02	0.03	0.04
ComplEx	8,622.68	0.23	0.17	0.25	0.33

Table 5.5: Best results of the two tested KGE algorithms. The table reports the Mean Reciprocal Rank (MRR), Mean Rank (MR) and hits at n score, i.e. how many elements of a vector of rankings make it to the top n positions.

Both embedding models had their best performances with a learning rate of 10^{-4}, and a Multiclass NLL Loss. The TransE model values were generated with 64 batches, 100 epochs, $k = 200$, 5 eta, and a regularizer Lp1 with $\lambda = 10^{-5}$. The ComplEx model values were generated with 128 batches, 250 epochs, $k = 150$, 20 eta, and a regularizer Lp3 with $\lambda = 10^{-5}$.

The best ComplEx model was selected for our final KGE, and its embedding is shown in Figures 5.17 and 5.18. Both figures are a graphical representation of the entities embedding in a $k = 150$ space, projected in a 2-D space, with the t-SNE algorithm, Figure 5.17, and with PCA, Figure 5.18.

Other possible research involve investigating further embedding algorithms and a mixture of embedding models (like ConvE with RotatE).

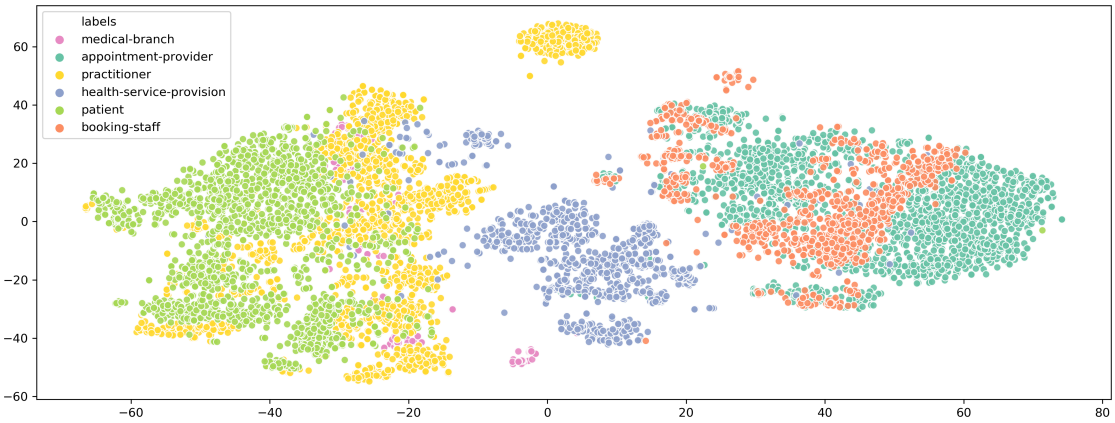


Figure 5.17: 2-D representation of the embedding space through the t-SNE algorithm. The colored points represent the different entities of the KG.

The t-distributed Stochastic Neighbor Embedding (t-SNE) is able to show the structure of the points in a low dimensional space, by bringing close points of the input space to neighbour points in the final space. It shows the three main groups of the health service booking system. The practitioner related area on the left that groups practitioners patient and the medical branch. The booking system with the appointment provider and booking staff on the right. And the health-services at the centre in-between these two systems.

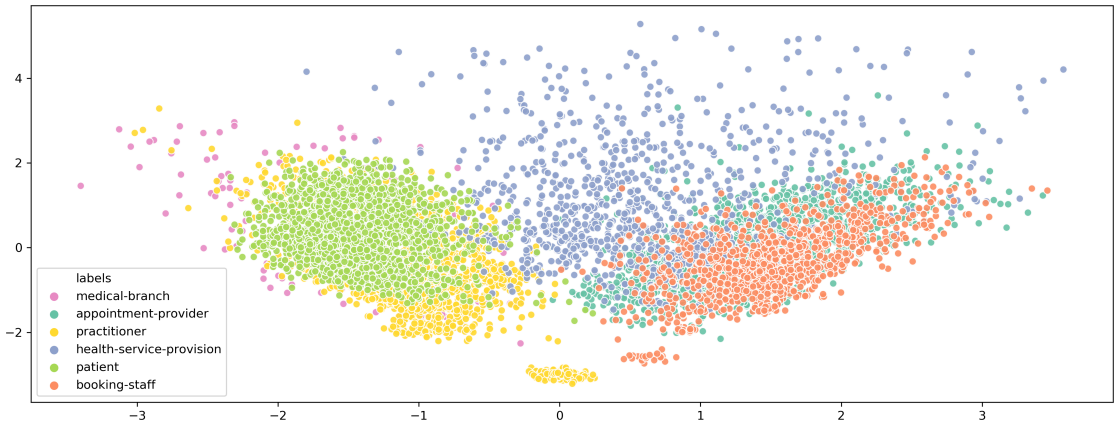


Figure 5.18: 2-D representation of the embedding space through PCA. The colored points represent the different entities of the KG by using the same color palette in Figure 5.17. The Principal Component Analysis (PCA) is a dimensional reduction technique that preserves large pairwise distances. In this image, on the x-axis the varies from practitioner-patient relation on the left, to the healthcare booking systems on the right. The y-axis tries to take into account more medical aspects of differences among health-services relations.

Visualization: embedding space To illustrate this embedding space, we used two separate techniques, t-SNE and PCA, to project all of the entities to a two-dimensional space. The t-SNE algorithm was designed to group nearby items together and separate them from distant elements. Instead, the PCA algorithm is an orthogonal transformation that focuses on the primary principal components that cause the most dispersion in the data. The entities are all colored according on their type. These visualizations are particularly fascinating since they illustrate that the entities are well partitioned and grouped by the relations defined in the schema. The referral relation connects patients, practitioners, and medical-branches, the reservation relation connects booking-staff and appointment-providers, and health-service-provisions are in the middle because they are in both relations.

5.3.4 Unsupervised learning results

As a preliminary and fundamental work, the number of clusters (K) has been determined. The maximum value in the range where the K value was picked is K=88 because this number is nearly equal to the square root of the number of elements clustered. The graphic shows a bar plot generated by our approach for determining the K value. Each Hierarchical Clustering (with a given linking mechanism) and K-Means were analyzed with 11 different indices for every K till K=88, and the best K values (as local maximum or local minimum depending on the index) are visualized by each cluster-index. Finally, the top selections based on the voting technique are K=13 and K=17. Following a preliminary investigation of both choices with the assistance of domain experts, we chose K=13 for our study. To retrieve non-trivial insights from the KGE, an unsupervised learning approach through clustering algorithms have been adopted. The main goal was to find useful and hidden groups of different entities which could explain and assess some hidden situations/behaviours inside the public healthcare system. Since the KGE involves different entities, such as patient, practitioner, provision and medical staff, discovering hidden relations among such entities can represent an added value for the stakeholders. The choice of the number of clusters (K), as the preliminary and fundamental task, was performed with the same procedure as reported in a previous work of the authors[23;106]. The maximum value of the range where the K value was chosen is $K = 88$ since this number is approximately equal to the square root of unique elements occurrence on which the clustering was carried out. It can be further remarked that the suggested K values are $K = 13$ and $K = 17$, as it can be seen in Figure 5.19. For our study, we selected $K = 13$ after a preliminary analysis of both the options aided by domain experts. Furthermore, we also investigated the DBscan clustering algorithm, looking for good input hyper-parameters. However, unfortunately, the obtained clusters were not promising in terms of results, and therefore we used the K-Means and Hierarchical Clustering algorithms, with the

four linkage methods: single, complete, average, and Ward.

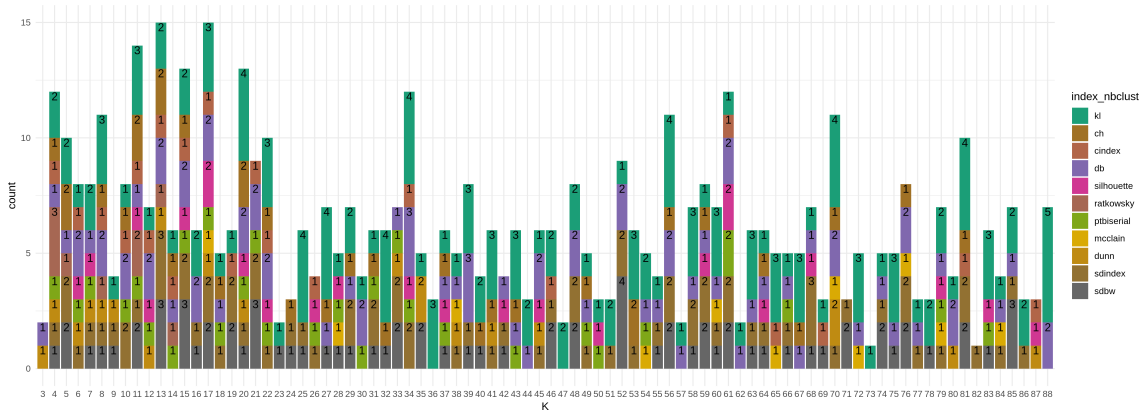


Figure 5.19: K-selection on KG. Bar plot obtained from the procedure described in[23;106] to choose the K value. Each Hierarchical Clustering (with a specified linkage method) and K-Means was evaluated with 11 different quality-indexes for every K until $K = 88$. By fixing a clustering, the value of an index varies as K varies. Given that, for each index, the best value can be maximum or minimum depending on its mathematical formalization, each index expresses one or more vote on what K gives the best cluster. All these votes are gathered in this bar plot, and the K that results as the top voted are $K = 13$ and $K = 17$. The indexes used for assessing the quality of the clusters are the: KL index, Calinski and Harabasz (CH) index, C-Index, Davies and Bouldin (DB) index, Silhouette, Ratkowsky, point-biserial index, McClain and Rao index, Dunn index, SD validity index, and the SDbw validity index[25].

The retrieved distribution of the different entities composing the KGE is reported, for each cluster, in Figure 5.20. The obtained clusters appear to be strongly connected to the embedded space in Figure 5.18, displayed with the same colours. In particular, the entities *patient* (green) and *practitioner* (in yellow) with *medical-branch* (pink) appear to be well separated from the entities *booking-staff* (orange) and *appointment-provider* (dark-green), while the *health-service-provision* (blue) entity is distributed almost equally over these two zones. This aspect is also related to the structure of the KG, which contains *referral* and *reservation* as fundamental relations connecting the two main groups of entities already found. The reservation

is organized by booking staff that book health services for appointment providers. While, referrals are requests of health services for patients, written by practitioners that decide the medical branch membership of the health service. In the following paragraphs, we report some insights regarding two selected clusters (7 and 8) to assess and demonstrate the usefulness of our approach.

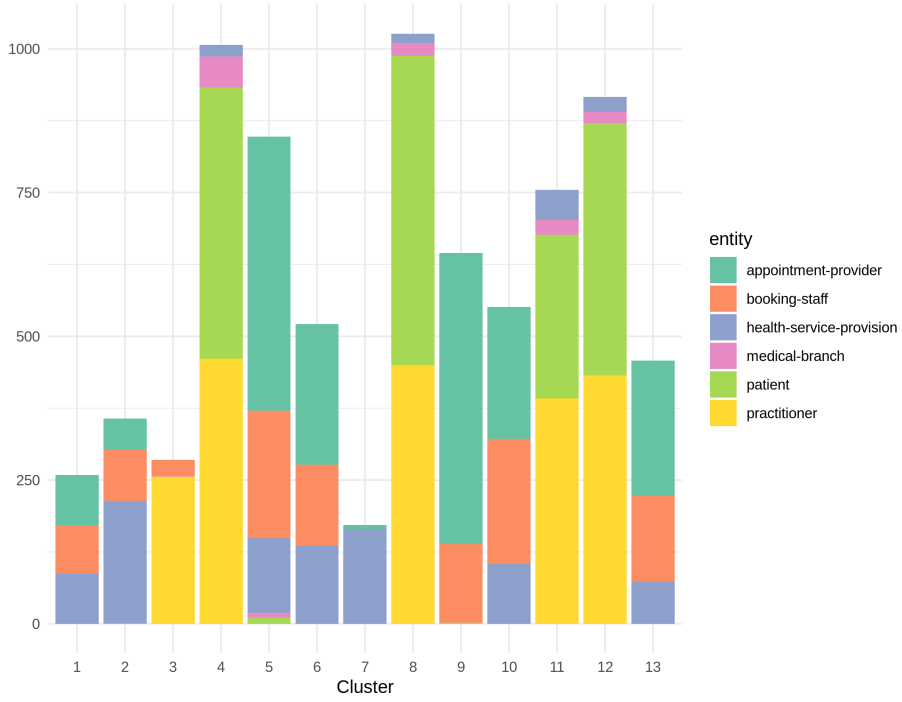
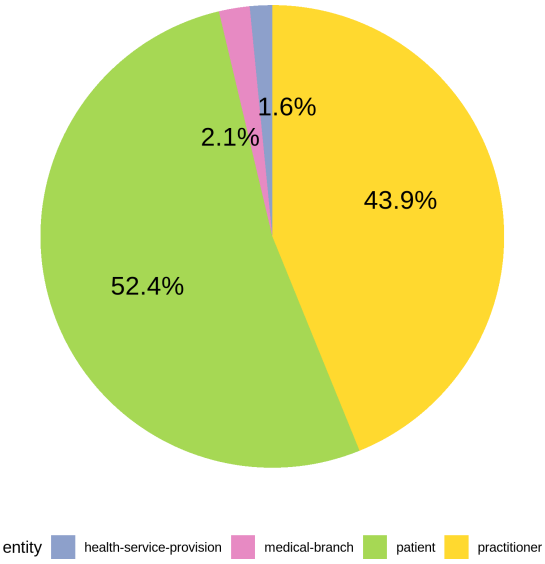


Figure 5.20: Bar plot of the size of the clusters colored by the entities obtained by fixing $K = 13$.

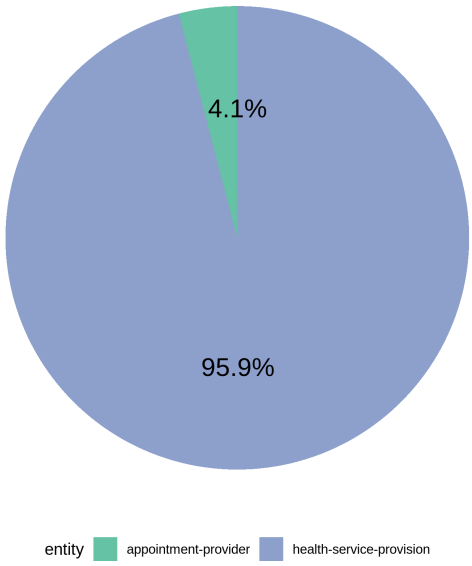
Clusters can be interpreted by looking at Figure 5.17, since they use the same colour palette. Clusters $1, 2, 5, 6, 7, 9, 10, 13$ encompass the booking aspects of the process, by combining appointment providers, booking staff and health services. Clusters $4, 8, 11, 13$ include mainly practitioner, patient, and medical branch, by allowing a small quota of health service, whereas cluster 3 includes practitioner and booking staff.

In Figures 5.21a and 5.21b the composition of clusters 7 and 8, in terms of the percentage of the presence of the entities, is shown.

To analyze in greater depth the different entities composing each cluster, in the following paragraph, we report some tables containing all the information about



(a) Pie chart of the distribution of cluster 8 by entity. This is one of the larger clusters, with slightly more than 1000 entities. It is composed by mainly patients (more than 50%), and practitioners. Still, there is a smaller part that adds up to 3.7% of the medical branch and health service entity.



(b) Pie chart of the distribution of cluster 7 by the entity. It is one of the smaller clusters, including less than 250 unique entity instances. Mostly health services provision make up the cluster, while slightly more than the 4% are related to appointment providers.

Figure 5.21: Pie chart of the distribution of cluster 7 and 8 by the entity.

attributes or entities linked to these cluster entities in the KG. Tables 5.6 and 5.7 report practitioner and patient of cluster 8 (Figure 5.21a), while Tables 5.8 and 5.9 present appointment providers and health service provisions from cluster 7 (Figure 5.21b). The tables report all the attributes that are linked to that entity. The numerical attributes are summarized with the use of basic statistics while the categorical attributes and other linked entities are presented as lists, sorted by frequency, in descending order.

First insights can be observed in cluster 8. Here, it is visible the strong relation among *practitioner* and *patient* entities: their descriptive tables 5.6 and 5.7 appear to be very similar. Table 5.6 reports the age of the practitioner's patient that varies on average within the range from 56 to 75 years, and in table 5.7 the patient age ranges from 53 to 76, with the same median (second quartile) of 67 years old.

By looking at the NUTS code, the majority of the patients are inhabitants of the cities of Naples, Vico Equense, Torre Annunziata and Forio d'Ischia. They mainly use the public healthcare system for medical examinations, especially laboratory tests for generic check-ups, in facilities related to the *local-health-department* A and *local-health-department* C (a local-health-department can also be named ASL). Moreover, more than 75% of *waiting-days* are zeros.

	patient-age	res-waiting-days	first-res-waiting-days	last-reservation-change-date	referral-date	booked-date	reservation-date
mean	64.5	8.4	1.1	2016-02-12	2016-02-10	2016-02-20	2016-02-12
std	14.9	32.3	8.0				
min	4.0	-72.0	-255.0	2012-12-03	2006-01-13	2014-01-02	2012-12-03
25%	56.0	0.0	0.0	2015-04-15	2015-04-07	2015-04-20	2015-04-15
50%	67.0	0.0	0.0	2015-12-15	2015-12-07	2015-12-17	2015-12-15
75%	75.0	0.0	0.0	2017-02-08	2017-02-03	2017-02-15	2017-02-08
max	101.0	444.0	342.0	2017-12-29	2075-09-08	2018-12-27	2017-12-29

encrypted-nin-id	gender	priority-code	nuts-istat-code	booking-agent-id	updating-booking-agent-id
(1448589, 125)	(1.0, 19559)	(0.0, 5905)	(63049, 14442)	(4207, 20858)	(-1, 27271)
(320220, 100)	(2.0, 18315)	(4.0, 2180)	(63067, 1393)	(4323, 964)	(2746, 547)
(⋆, 777937, 99)		(3.0, 357)	(63083, 1323)	(2746, 550)	(2938, 488)
(976398, 97)		(2.0, 144)	(63086, 1139)	(2850, 512)	(4412, 481)
(⋆, 706294, 96)		(1.0, 45)	(63031, 1079)	(2938, 492)	(2824, 336)
(756362, 96)			(63023, 1018)	(4412, 481)	(3717, 317)
(1219303, 95)			(63059, 819)	(5522, 415)	(2850, 236)

referral-centre-id	number-of-health-services	local-health-department-id	refined-health-service-id
(ASC/70-ASC/70, 8871)	(1, 37155)	(A, 28433)	(PRELIEVO DI SANGUE VENOSO, 2227)
(LAB/EST-EST30, 3857)	(8, 282)	(C, 8108)	(EMOCROMO: Hb, GR, GB, HCT, PLT, IND. DERIV.,....
(SGB/LAB-ANASGB, 2499)	(2, 224)	(B, 1333)	(ALANINA AMINOTRANSFERASI (ALT) (GPT) [S/U], 1...
(PATCLINC-INC9, 2175)	(4, 118)		(ASPARTATO AMINOTRANSFERASI (AST) (GOT) [S], ...
(PATPEL-ANPPEL, 1975)	(3, 34)		(GLUCOSIO [S/P/U/dU/La], 1056)
(8282-8282, 1474)	(5, 16)		(COLESTEROLO TOTALE, 1000)
(572/24-572/24, 1100)	(7, 15)		(BILIRUBINA TOTALE E FRAZIONATA, 974)

Table 5.6: Summary of information about the *practitioners* belonging to cluster 8. An asterisk is present if the entity is itself present in the same cluster. The table reports the statistics of all the attributes that are linked to the practitioners in cluster 8. In this table, there is not *practitioner-id*, because we are reporting the properties of the practitioners of the cluster, and when there is an asterisk the instance is in the same cluster, and so there is a semantic adherence.
Here we can see, for example, that on average the patient age, treated by the practitioners, is 64.5 years old, and the health services that these practitioners tend to prescribe are in the area of blood test related. Furthermore, column *encrypted-nin-id* reports the matched ids of the patients, that are also part of the entities of Table 5.7.

	patient-age	res-waiting-days	first-res-waiting-days	last-reservation-change-date	referral-date	booked-date	reservation-date
mean	63.4	7.2	1.2	2016-03-19	2016-03-14	2016-03-26	2016-03-19
std	16.7	28.5	8.9				
min	1.0	-120.0	-133.0	2013-05-27	2013-05-24	2014-01-02	2013-05-27
25%	53.0	0.0	0.0	2015-04-01	2015-03-16	2015-04-10	2015-04-01
50%	67.0	0.0	0.0	2016-03-22	2016-03-04	2016-03-23	2016-03-22
75%	76.0	0.0	0.0	2017-03-30	2017-03-29	2017-04-07	2017-03-30
max	94.0	451.0	180.0	2017-12-29	2075-09-08	2018-08-30	2017-12-29

gender	priority-code	nuts-istat-code	practitioner-id	booking-agent-id	updating-booking-agent-id
(1.0, 18924)	(0.0, 6261)	(63049, 17227)	(-1, 3528)	(4207, 16415)	(-1, 26796)
(2.0, 18803)	(4.0, 2393)	(63031, 2133)	(⋆, 23909, 3393)	(3282, 1497)	(4412, 714)
	(3.0, 271)	(63086, 1502)	(⋆, 23886, 784)	(4323, 1381)	(2746, 572)
	(2.0, 138)	(63024, 1245)	(⋆, 23867, 609)	(5522, 1259)	(2824, 550)
	(1.0, 43)	(63071, 997)	(18964, 331)	(5403, 978)	(3116, 548)
		(63038, 938)	(24949, 330)	(4412, 714)	(2938, 547)
		(63072, 771)	(20035, 329)	(3116, 708)	(2887, 395)

referral-centre-id	number-of-health-services	local-health-department-id	refined-health-service-id
(ASC/70-ASC/70, 5036)	(1, 37085)	(A, 23218)	(PRELIEVO DI SANGUE VENOSO, 2154)
(LAB/EST-EST30, 3127)	(2, 234)	(C, 9218)	(EMOCROMO: Hb, GR, GB, HCT, PLT, IND. DERIV.,....
(572/24-572/24, 3116)	(8, 195)	(B, 5291)	(COLESTEROLO TOTALE, 1067)
(PATPEL-ANPPEL, 2793)	(4, 135)		(ALANINA AMINOTRANSFERASI (ALT) (GPT) [S/U], 1...
(SGB/LAB-ANASGB, 2155)	(3, 36)		(ASPARTATO AMINOTRANSFERASI (AST) (GOT) [S], ...
(PATCLINC-INC9, 1511)	(6, 11)		(TRIGLICERIDI, 980)
(8282-8282, 1343)	(10, 8)		(GLUCOSIO [S/P/U/dU/La], 945)

Table 5.7: Summary of information about the *patients* belonging to cluster 8. An asterisk is present if the entity is itself present in the same cluster. This table is complementary to Table 5.6. Here are the statistics of all the entity/attribute related to the patients in cluster 8, e.g the average age of these patients is 63.4 years old, the most common health services that the patients do is blood test related, and the practitioners that have in cure the patients in cluster 8 are mostly the practitioner that are within cluster 8, as shown by the asterisks in the *practitioner-id* column. Note that the ids with -1 represent anonymous practitioners or booking agents.

Other insights can be obtained from observing cluster 7, in Figure 5.21b. It presents a high predominance of the *health-service-provisions* entity. It contains only laboratory tests for check-ups as medical services, these representing the majority of the examinations performed. As can be seen from Tables 5.8 and 5.9, all the *appointment-providers* of this cluster belong to the *local-health-department* A. Moreover, it can also be noticed that the medical examinations inside this cluster are, in general, services provisions carried out by *appointment-providers* related to all the *local-health-departments*. Yet this cluster contains only *appointment-providers* of *local-health-department* A. Most of the patients that go to *appointment-providers* of *local-health-department* A are from Naples, Vico Equense, Forio, Torre del Greco and Torre Annunziata. Plus, the average value of *waiting-days* relating to medical services carried out by facilities of *local-health-department* A is 0, while, for the other two *local-health-department*, this value is slightly higher. Finally, other differences among Tables 5.8 and 5.9 account for the priority code and a quite different composition of booking agents, especially the updating ones. These different values might be due to the fact that the structures in A are bigger and

better organized, and the updating booking agents with a diverse mix of priority codes might help in this process.

In summary, unsupervised learning approaches, such as clustering, applied to KGE, have allowed us to reveal hidden information that would have been more complex to infer through the original legacy dataset structure. The reported information is aggregated statistical proprieties of a subset of the patient's age, specific codes area, practitioners, health services, health providers, booking agents and waiting time, all of them related to each other by the generated paths. From a health-service booking manager prospective, knowing those links and aggregate statistics could help to decide where to invest for improvements in the service quality, assessed by a variety of key performance indicator as waiting time.

	patient-age	res-waiting-days	first-res-waiting-days	last-reservation-change-date	referral-date	booked-date	reservation-date
mean	62.1	0.0	0.0	2015-11-18	2015-11-13	2015-11-18	2015-11-18
std	14.9	0.0	0.0				
min	1.0	0.0	0.0	2014-01-02	2006-01-13	2014-01-02	2014-01-02
25%	53.0	0.0	0.0	2015-01-15	2014-11-21	2015-01-15	2015-01-15
50%	65.0	0.0	0.0	2015-10-21	2015-10-14	2015-10-21	2015-10-21
75%	72.0	0.0	0.0	2016-11-10	2016-11-02	2016-11-10	2016-11-10
max	97.0	6.0	3.0	2017-12-29	2075-09-08	2017-12-29	2017-12-29

encrypted-nin-id	gender	priority-code	nuts-istat-code	practitioner-id	booking-agent-id	updating-booking-agent-id
(868291, 103)	(1.0, 25421)	(4.0, 266)	(63049, 32388)	(23909, 9136)	(4207, 42428)	(-1, 49525)
(1248640, 100)	(2.0, 24810)	(0.0, 240)	(63067, 1474)	(-1, 2452)	(4323, 4301)	(5103, 652)
(818756, 100)			(63023, 1153)	(23886, 2278)	(4217, 1073)	(3759, 33)
(757787, 100)			(63059, 1096)	(24949, 814)	(5103, 652)	(4693, 18)
(1248874, 99)			(63089, 771)	(23930, 497)	(4246, 576)	(3765, 1)
(900726, 98)			(63002, 668)	(24125, 473)	(4241, 470)	(3891, 1)
(756362, 97)			(63064, 644)	(24153, 391)	(4210, 151)	(3788, 1)

number-of-health-services	local-health-department-id	refined-health-service-id
(1, 49566)	(A, 50231)	(PRELIEVO DI SANGUE VENOSO, 2634)
(2, 368)		(⋆, EMOCROMO: Hb, GR, GB, HCT, PLT, IND...
(4, 149)		(⋆, ASPARTATO AMINOTRANSFERASI (AST) (G...
(3, 83)		(⋆, ALANINA AMINOTRANSFERASI (ALT) (GPT)...
(8, 50)		(⋆, GLUCOSIO [S/P/U/dU/La], 1918)
(5, 9)		(⋆, COLESTEROLO TOTALE, 1713)
(7, 5)		(UREA [S/P/U/dU], 1710)

Table 5.8: Summary of information about the *appointment-providers* belonging to cluster 7. An asterisk is attached to the health service in the same cluster, i.e. it belongs to the entities whose attributes are synthesized in Table 5.9. It is worth to notice that this table is the only one that includes a smaller subset of priority codes and one local health department.

	patient-age	res-waiting-days	first-res-waiting-days	last-reservation-change-date	referral-date	booked-date	reservation-date
mean	63.8	1.4	0.1	2016-04-03	2016-03-28	2016-04-04	2016-04-03
std	16.0	10.2	0.7				
min	1.0	-219.0	-42.0	2013-09-05	2006-01-13	2014-01-02	2013-09-05
25%	55.0	0.0	0.0	2015-05-28	2015-05-07	2015-05-28	2015-05-28
50%	67.0	0.0	0.0	2016-05-11	2016-04-07	2016-05-11	2016-05-11
75%	75.0	0.0	0.0	2017-03-28	2017-03-22	2017-03-29	2017-03-28
max	101.0	366.0	82.0	2017-12-29	2075-09-08	2018-03-19	2017-12-29

encrypted-nin-id	gender	priority-code	nuts-istat-code	practitioner-id	booking-agent-id	updating-booking-agent-id
(756362, 86)	(2.0, 39855)	(0.0, 10399)	(63049, 33979)	(23909, 7965)	(4207, 40924)	(-1, 62510)
(986711, 85)	(1.0, 39054)	(4.0, 3738)	(63086, 4034)	(-1, 6666)	(4323, 3212)	(2938, 1529)
(1248640, 84)		(3.0, 493)	(63031, 3775)	(23886, 1966)	(3282, 2984)	(2746, 1259)
(754507, 84)		(2.0, 197)	(63071, 1977)	(23867, 891)	(5522, 2464)	(2824, 1105)
(637389, 84)		(1.0, 41)	(63038, 1852)	(24949, 643)	(5403, 1687)	(4412, 936)
(1254488, 82)			(63084, 1812)	(20029, 581)	(2938, 1533)	(3116, 900)
(817583, 81)			(63083, 1810)	(14585, 521)	(3116, 1449)	(2887, 859)

referral-centre-id	number-of-health-services	local-health-department-id
(⋆, ASC/70-ASC/70, 12878)	(1, 77883)	(A, 49124)
(⋆, LAB/EST-EST30, 7871)	(2, 371)	(C, 20301)
(572/24-572/24, 6474)	(8, 256)	(B, 9484)
(⋆, PATPEL-ANPPEL, 6045)	(4, 252)	
(⋆, SGB/LAB-ANASGB, 4982)	(3, 110)	
(8282-8282, 4389)	(5, 17)	
(PATCLINC-INC9, 4342)	(7, 12)	

Table 5.9: Summary of information about the *health-service-provisions* belonging to cluster 7. An asterisk is present if the entity is itself present in the same cluster. Despite having some similar attributes statistics as Table 5.8, like a balanced gender,common referral centers and health services, the two tables differentiate from the local health department, the waiting days, priority code and booking agent, in particular the updating one.

entities distribution Let's look at the clustering obtained with K=13. The recovered distribution of the various entities constituting the clusters is depicted in this figure. The obtained clusters appear to be closely related to the embedded space seen on the left. As previously stated, this occurrence is strongly tied to the Knowledge Graph schema, which has two basic relations connecting the two entity groups discovered.

So, in order to examine patients in cluster number 8, we report all the information about qualities or entities associated to cluster's patients in the Knowledge Graph in these tables. In particular, numerical attributes are summarized using basic statistics (such as mean, standard deviation, percentiles, and so on), whereas categorical attributes and other connected entities are given as lists, sorted descendingly by frequency. For example, the mean age of all patients in this cluster is 63, indicating that the vast majority of them are elderly individuals. The mean number of waiting days for a single reservation is 7 days, while the 75th percentile is 0, indicating that the mean is skewed toward a high number, implying that there are some outliers with a very high number of waiting days, such as the maximum 451 (451 days). For a categorical feature such as nuts-istat-code, the most frequent values are displayed along with the number of repetitions. We can see that the majority of the patients in this cluster are from the Naples area code 63049. Except for the missing practitioner, the most recurring practitioners associated to these patients are those who are also inside the cluster, as indicated by the star to the left side of each practitioners-id. The most frequent appointment-providers are those from Naples's center who perform laboratory tests for check-ups. Indeed, these examinations are among the most used health-care services.

Cluster 8 on Grakn Some typical and valuable tasks may be readily completed by using Grakn to describe and manage our Knowledge Graph, due to the supplied API, the Graql query language, and the Workbase (the graphical interface). Some results have also been produced using the Graql programming language. It has proven feasible to deduce implicit information using such traits. This diagram displays Cluster 8 as well as the connections that two patients have with practitioners in the same cluster. There are significantly more connections between them than there are between other patients and practitioners. This simple query found the links in the Grakn Workbase.

5.3.5 Insights data visualization

In selecting Grakn to represent and manage our KG, some common and useful tasks can be easily accomplished thanks to the provided API, Graql (Grakn's analytics query language) and the Workbase (the graphical interface). With Grakn it is possible to aggregate values over the dataset, such as `count` or `median` values as well as common query patterns, with a conjunction, disjunction or negation. Some results have also been obtained by using the Graql language and Python API. In this paragraph, we show an example of the graphical interface for the data visualization. Thanks to such features, it has been possible to infer implicit information. Figure 5.22 depicts the cluster 8 and the links that two *patients* have with the same *practitioner*, and highlights that there are many more links between those two *patients* in comparison with links among other represented entities.

5.3.6 Supervised learning on KG

As the last step of the pipeline, we provided a *state-of-the-art* predictive framework based on supervised learning techniques, as described in Section 4.1.3, to explore the possibility of predicting whether or not a patient will have at least one medical examination, in terms of clusters, during the year 2018 (the last year available in our dataset).

In a preliminary phase, we exploited the KG to group the *health-service-provisions*, since taking them separately would generate matrices that would be too sparse. Moreover, this allowed us to use the embedding triples obtained with the patients themselves as a feature for the predictive algorithms (ignoring those related to the year 2018). The selected clusters are the ones that contain at least one *health-service-provision* and these provisions have to be present in the year 2018 with a threshold set to 1% of the total number of provisions in the year 2018.

To evaluate the performance of the proposed supervised algorithms, the dataset has

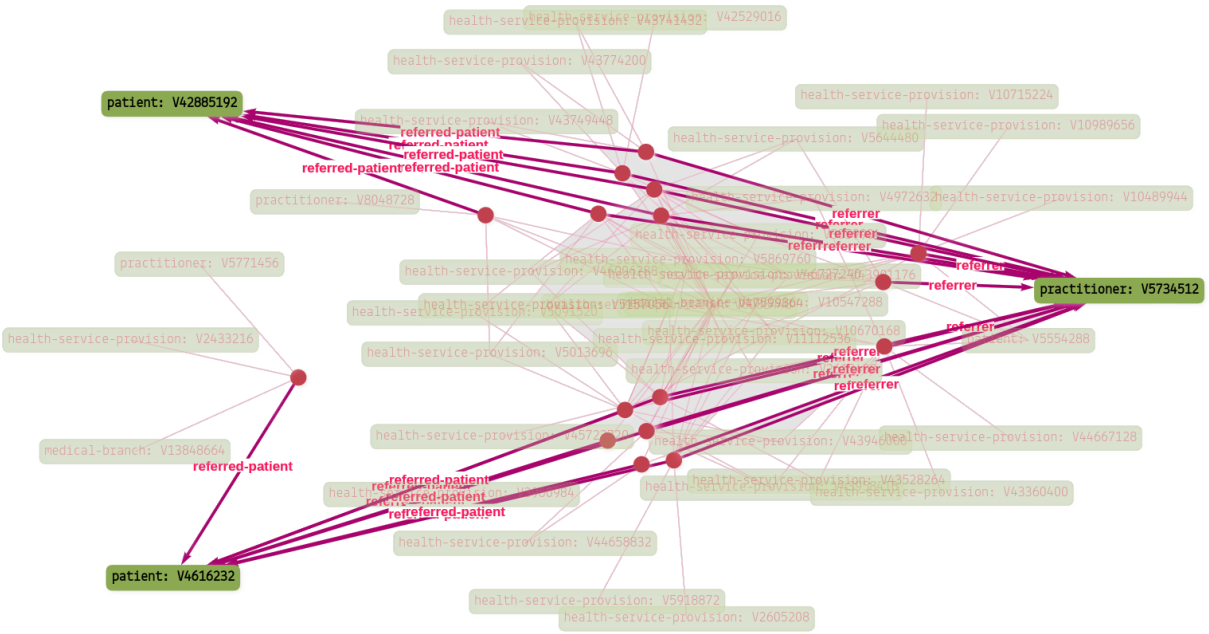


Figure 5.22: Query result from the Grakn Workbase. This image shows an example from cluster 8. Two *patients* and their common *practitioner* are highlighted with respect to the other surrounding links.
The image shows that in this environment once the cluster is defined, a healthcare manager can query exactly that information resulting in a link according to the cluster.
Magenta circles represent meta-relations, rounded green rectangles are entities. In this case, the meta-relation is the referral, that connects a doctor with its patient, and the prescribed health services. In fact, from these circles, three kinds of lines branch out: the *referrer* lines connecting a referral to a practitioner, *referred-patient* lines connecting referral to a patient, the lines in the background that connect referrals to a health service.
The links were obtained by querying the KG for specific patient-practitioners, identified by their ids, reported in cluster 8; the two patients are the ones that appear in Table 5.6 with ids 777937 and 706294, while the practitioner id the one identified as 23867 in the column `practitioner-id` in Table 5.7 and Table 5.9.

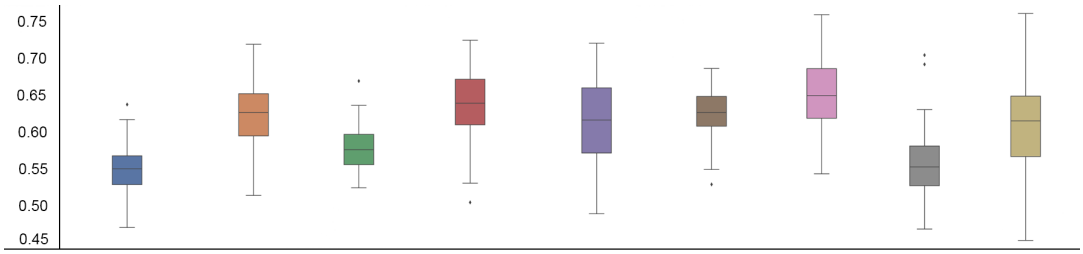
been divided as follows: 80% as the training set and 20% as the test set. Moreover, the vectors belonging to each of these sets were randomly selected. This task was repeated for each cluster that we analyzed. Each algorithm was executed 30 times using random permutation cross-validation, and the results were evaluated with the *Areas Under the Curve* criteria (AUC). This measure requires no threshold to be fixed and the results are more reliable than the *accuracy* when dealing with unbalanced datasets[74]. The AUC is calculated by plotting the true positive rate (recall) and false positive rate as the threshold is changed and finally calculating the area under this curve.

From Table 5.10 it can be observed that the Naive Bayes algorithm outperforms the other classifiers, reaching very high values with clusters 6 and 11. Significant results were also achieved with Gaussian Process and Linear SVM.

Cluster	Nearest Neighbour	Linear SVM	RBF SVM	Gaussian Process	Random Forest	Neural Network	Naive Bayes	QDA	XGBoost
10	0.59 ± 0.024	**0.67 ± 0.027**	0.58 ± 0.025	**0.67 ± 0.03**	0.62 ± 0.024	0.64 ± 0.028	0.65 ± 0.027	0.63 ± 0.02	0.65 ± 0.022
5	0.51 ± 0.056	0.58 ± 0.065	0.54 ± 0.079	0.6 ± 0.093	0.57 ± 0.071	0.56 ± 0.075	**0.64 ± 0.082**	0.52 ± 0.098	0.5 ± 0.079
4	0.54 ± 0.037	0.58 ± 0.045	0.55 ± 0.042	**0.65 ± 0.045**	0.58 ± 0.046	0.6 ± 0.051	0.63 ± 0.035	0.51 ± 0.051	0.57 ± 0.041
2	0.51 ± 0.029	0.54 ± 0.028	0.52 ± 0.036	0.56 ± 0.037	0.53 ± 0.05	**0.59 ± 0.031**	0.56 ± 0.036	0.49 ± 0.048	0.57 ± 0.031
1	0.54 ± 0.031	0.6 ± 0.044	0.6 ± 0.034	0.59 ± 0.041	0.6 ± 0.037	**0.62 ± 0.044**	0.61 ± 0.036	0.5 ± 0.055	**0.62 ± 0.033**
7	0.59 ± 0.022	**0.67 ± 0.021**	0.57 ± 0.021	0.66 ± 0.025	0.62 ± 0.021	0.62 ± 0.024	0.65 ± 0.022	0.58 ± 0.022	0.62 ± 0.021
6	0.57 ± 0.031	0.69 ± 0.025	0.59 ± 0.028	0.71 ± 0.028	0.69 ± 0.027	0.67 ± 0.022	**0.72 ± 0.024**	0.54 ± 0.026	0.71 ± 0.026
11	0.52 ± 0.04	0.65 ± 0.049	0.62 ± 0.038	0.68 ± 0.068	0.68 ± 0.046	0.69 ± 0.055	**0.73 ± 0.052**	0.52 ± 0.044	0.69 ± 0.052
13	0.59 ± 0.02	**0.65 ± 0.026**	0.57 ± 0.025	0.64 ± 0.041	0.62 ± 0.027	0.64 ± 0.028	0.64 ± 0.025	0.62 ± 0.022	0.64 ± 0.021
MEAN	0.55 ± 0.034	0.63 ± 0.052	0.57 ± 0.031	0.64 ± 0.048	0.61 ± 0.051	0.63 ± 0.04	**0.65 ± 0.052**	0.55 ± 0.052	0.62 ± 0.065

Table 5.10: AUC values of each classification technique, repeated 30 times, for each selected cluster written in the form of (mean ± std). In the last row, the mean values of each classification technique are recorded. The values in bold are the best values for each row. In our tests Naive Bayes performed better than other techniques.

Supervised Learning As the pipeline's final phase, we developed a predictive framework based on supervised learning techniques to forecast whether or not a patient will undergo at least one of the medical examinations, in terms of clusters, during 2018. (the last year available in our data set). Initially, we used the Knowledge Graph to group the health-service-provisions because doing so independently would result in sparse matrices. Furthermore, it enabled us to use the embedding space for the patients as a feature for predictive algorithms



Cluster	Nearest Neighbors	Linear SVM	RBF SVM	Gaussian Process	Random Forest	Neural Network	Naive Bayes	QDA	XGBoost
10	0.59 ± 0.024	**0.67 ± 0.027**	0.58 ± 0.025	**0.67 ± 0.03**	0.62 ± 0.024	0.64 ± 0.028	0.65 ± 0.027	0.63 ± 0.02	0.65 ± 0.022
5	0.51 ± 0.056	0.58 ± 0.065	0.54 ± 0.079	0.6 ± 0.093	0.57 ± 0.071	0.56 ± 0.075	**0.64 ± 0.082**	0.52 ± 0.098	0.5 ± 0.079
4	0.54 ± 0.037	0.58 ± 0.045	0.55 ± 0.042	**0.65 ± 0.045**	0.58 ± 0.046	0.6 ± 0.051	0.63 ± 0.035	0.51 ± 0.051	0.57 ± 0.041
2	0.51 ± 0.029	0.54 ± 0.028	0.52 ± 0.036	0.56 ± 0.037	0.53 ± 0.05	**0.59 ± 0.031**	0.56 ± 0.036	0.49 ± 0.048	0.57 ± 0.031
1	0.54 ± 0.031	0.6 ± 0.044	0.6 ± 0.034	0.59 ± 0.041	0.6 ± 0.037	**0.62 ± 0.044**	0.61 ± 0.036	0.5 ± 0.055	**0.62 ± 0.033**
7	0.59 ± 0.022	**0.67 ± 0.021**	0.57 ± 0.021	0.66 ± 0.025	0.62 ± 0.021	0.62 ± 0.024	0.65 ± 0.022	0.58 ± 0.022	0.62 ± 0.021
6	0.57 ± 0.031	0.69 ± 0.025	0.59 ± 0.028	0.71 ± 0.028	0.69 ± 0.027	0.67 ± 0.022	**0.72 ± 0.024**	0.54 ± 0.026	0.71 ± 0.026
11	0.52 ± 0.04	0.65 ± 0.049	0.62 ± 0.038	0.68 ± 0.068	0.68 ± 0.046	0.69 ± 0.055	**0.73 ± 0.052**	0.52 ± 0.044	0.69 ± 0.052
13	0.59 ± 0.02	**0.65 ± 0.026**	0.57 ± 0.025	0.64 ± 0.041	0.62 ± 0.027	0.64 ± 0.028	0.64 ± 0.025	0.62 ± 0.022	0.64 ± 0.021
MEAN	0.55 ± 0.034	0.63 ± 0.052	0.57 ± 0.031	0.64 ± 0.048	0.61 ± 0.051	0.63 ± 0.04	**0.65 ± 0.052**	0.55 ± 0.052	0.62 ± 0.065

Figure 5.23: SupervidedAUC

(ignoring the ones related to the year 2018). The selected clusters have at least one health-service provision, and these provisions must be present in 2018 in more than a threshold established at 1% of the overall number of provisions in 2018. To assess the performance of the suggested supervised algorithms, the data set was divided as follows: 80% as a training set and 20% as a test set. Furthermore, the items in each of these sets were chosen at random; this operation was done for each cluster we examined.

AUC Each method was run 30 times using random permutation cross-validation, and the results were assessed using the Areas Under the Curve criteria (AUC). When dealing with unbalanced datasets, this metric has no preset threshold and produces more dependable findings than Accuracy. In this table, the AUC values for each classification technique are written in the form of (mean std) and are repeated 30 times for each selected cluster. The mean values of each classification approach are shown in the last row. The values in bold are the best values for each row. The Naive Bayes algorithm beats the other classifiers in this table, achieving very high values with clusters 6 and 11. The Gaussian Process and Linear SVM also produced good results.

Chapter 6

Biosensor application

Creating a portable system to detect heavy metal contamination and diagnose on-site would benefit society. In many real-world applications, a smartphone could eventually act as a substance concentration detector. A 3D printed optical interface typically extends the camera's capabilities on smartphones.

This chapter presents a Proof of Concept Machine Learning architecture that can automatically identify wells in a multiwell and predict fluorescence intensity without an external instrument. The first stage focuses on image processing and feature extraction using a scale-invariant feature transformation, while the second stage combines three machine learning algorithms (Multi-layer Perceptron, Random Forest, and XGBoost) for classification and prediction. In the presence of a specific protein, this architecture measures the fluorescence of a water assay containing Mercury. It recognizes wells in three different light environments: ambient light, portable UV light, and homogeneous diffusive fixed UV light. This proof-of-concept can be applied in a general framework for feature-based image knowledge extraction.

6.1 Extraction of Keypoints and descriptors

To recognize an object in a photo, it is possible to rely on an algorithm called SIFT, scale-invariant feature transformation, in our expert system solution. During the development process, we tested multiple techniques to extract ROI from well plates, among all we analyzed the segmentation techniques proposed in[3;122]. In particular we tested k-means clustering of Steinley and Brusco[131], and edge detector for contouring by Canny[22], applied on multiple combinations of color spaces channels extracted from the images, encompassing RGB (Red, Green, Blue), LAB (lightness, green-red, blue-yellow), and HSV (hue, saturation, value). Eventually we used SIFT.[75] proposed a SIFT algorithm that is invariant to translation, scale, and rotation, and is robust to affine distortion, change in lighting, and change in 3D point of view. It can address the problem of recognizing a 3D object photographed from different points of view and light. The SIFT algorithm identifies and describes keypoints, i.e.points in a picture that are interesting or stand out. In our context, SIFT generate is a set of key point-descriptor, (p, s, r, f), where $p = (x, y)$ is the location of the key point pixel on the image, s the scale, r the orientation, and f a 128-dim descriptor generated from local image gradients. SIFT algorithm can be sensitized in four main steps described in the following.

STEP 1. Keypoints detection in scale space

Given the image $I(x, y)$, with x, y location coordinates, and G the Gaussian Blur operator

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x+y)^2/2\sigma^2},$$

The SIFT algorithm starts by identifying the candidate keypoint positions as the local extreme in a difference-of-Gaussian pyramid that approximates the second-

order derivatives of the scale space of the image[73]. For each octave $o \in O$ the linear scale-space representation of an image signal, $L_o(x, y, \sigma)$ is defined by $L_o(x, y, 0) = I(x, y)$ and

$$L_o(x, y, \sigma_i) = G(x, y, \sigma_i) * I(x, y),$$

where σ_i is the quantised scale parameter, and the convolution, $*$, *blurs* the image I. To create each octave, we first need to choose the number of images we want in each octave. This is denoted by S. Then the σ for the Gaussian filter is chosen to be multiplied by $k = 2^{(1/S)}$. Since blur accumulates multiplicatively, when we blur the original image with this filter s times, the result will have $\sigma_S = 2\sigma_0$. Moreover, since each octave's image size is half the previous one:

$$L_o(x, y, \sigma) = L_{o-1}(\frac{x}{2}, \frac{y}{2}, \sigma)$$

Regarding the parameters,[75] proposes as optimal values: $O = 4, S = 5$ and base scale level $\sigma_0 = 1.6$. The value of σ, at a given octave o and scale level s is given by:

$$\sigma_{s+o} = \sigma_s(o) = \sigma(o, s) = \sigma_0 2^{o+\frac{s}{S}}, \qquad s \in [0, S-1], o \in [0, O-1].$$

The SIFT algorithm uses the blurred images of the Gaussian pyramid at this stage to produce another collection of images, the Difference of Gaussians (DoG) pyramid. In an octave, two consecutive images are chosen and one is subtracted from the other, then the next consecutive pair is taken, and this is repeated for all octaves. The images resulting from this method, the DoGs, are actually an approximation of the Laplacian operator on each octave of the image's scale-space representation:

$$D_o(x, y, \sigma_i) := L_o(x, y, \sigma_{i+1}) - L_o(x, y, \sigma_i) \approx (k-1)(k^i\sigma_0)^2\nabla^2 L_o,$$

where $k = 2^{(1/S)}$. Note that the generation of an extra level, $S + 1$ is needed by the algorithm because there will be one less image than in the expected octave when adjacent levels are subtracted. In addition, in the search for locally distinct points, this only preserves the frequencies between the blur level of both images when subtracting differently blurred images from each other. This technique filters out high-frequencies (noise) but preserves local differences; so it acts as a band-pass filter. Finally, when there are edges in the image, i.e. a line across which there is a shift of intensity, the magnitude of the gradient ∇L_o will be independent of the scale σ at the location of the edge because we are using normalized derivatives.

STEP 2. THE KEYPOINT FILTERING

Once candidate keypoints are placed in scale space by comparing pixels to their neighbor, the number of interpolated keypoints is limited, so to maximize performance and robustness. The function value at the extremum of the DoG octave expansion of the Second-order Taylor, developed by[17], is used for rejecting the low-contrast extreme by fixing the intensity scale-space value threshold of the subpixel. In the difference-of-Gaussian equation, a poorly defined peak would have a broad primary curvature around the edge but a small one in the perpendicular direction: thus the high edge response is determined based on the ratio of principal curvatures, as the principal curvatures can be calculated from a 2x2 Hessian matrix. In conclusion, after eliminating any low-contrast keypoints and edge keypoints; what remains are strong interest points. At this stage of the algorithm, with respect to the final output (p, σ, r, f), the output generated so far will contain the set of keypoints (p, σ) where p identifies the location of the keypoints of the image and σ the level of scale at which is located the keypoint.

STEP 3. the Keypoint Orientation Assignment

Translational, scale, and rotational invariance is achieved by SIFT, by convolution filters, scale invariance by scale-normalized DoG, while rotational invariance is achieved by angle histograms. Depending on the scale, an orientation corresponding to its dominant gradient direction is assigned around the neighborhood of each keypoint location. The magnitude and angle of a pixel gradient is determined and this allows to construct a histogram based on 36-bin angles (10 degrees for each bin). The direction of the keypoint will be identified with the maximum bin histogram. This direction will give the triplet to identify the keypoint, and will be used to normalize the descriptors, calculated in the last step.

STEP 4. Keypoint Descriptor Generation

The final stage of the SIFT algorithm is to generate a keypoint *descritor* that summarizes around that point the local structure. The keypoint descriptor is created by selecting 16×16 window across the keypoint, and this 16×16 window is then split into 8 4×4 windows. Gradient magnitudes and orientations are computed within each 4×4 window. These orientations are inserted into an 8 bin histogram (45 degrees each), to create a $4 \times 4 \times 8 = 128$ descriptor vector from each keypoint. In order to achieve rotation independence, the rotation of the keypoint is subtracted from each orientation, while a threshold, for illumination independence, is applied to minimize the influence of large gradient values.

6.2 Image analysis

According to[120] fluorescence-activated cell sorting, first reported in Science 52 years ago, has transformed biomedical research by allowing to isolate cells based on the expression of labeled proteins. Flow cytometric cell sorting, on the other

hand, has been blind to spatial processes, which is traditionally measured using microscopy.[120] combined ultrafast microscopy and image analysis with a flow cytometric cell sorter to unlock spatial phenotype. The authors demonstrate how this technology can be used to rapidly isolate cells with complex cellular phenotypes and to speed up genome-wide microscopy-based CRISPR screening, as shown in Figure 2.5

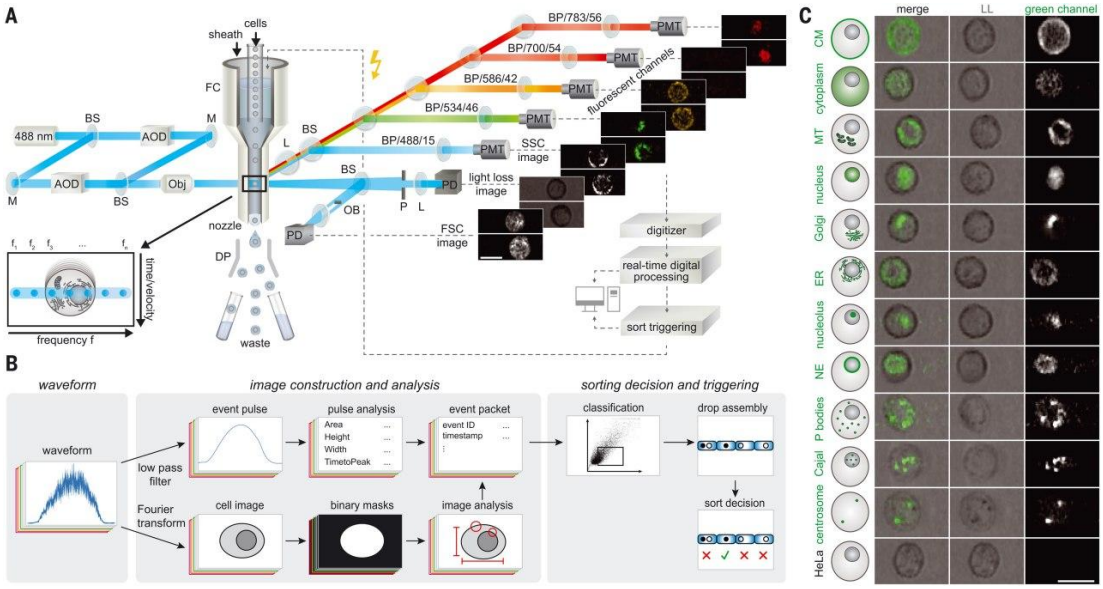


Figure 6.1: image analysis with a flow cytometric cell sorter to unlock spatial phenotype. Credits[120].

Developing a simple and compact system for detecting heavy metal contamination and providing on-site diagnoses would benefit society. Physical detection instruments are often connected to a smartphone to collect the most information from the assay, and a mobile device could eventually act as a detector of a substance concentration in many real-world applications. Typically, smartphone-based solutions include building a 3D printed optical interface to expand the camera's capabilities. This study presents a Proof of Concept Machine Learning architecture to assist natural scientists in their daily assay research by automatically reconsigning wells of a multiwell and predicting the fluorescence intensity without the need for an external instrument.

6.2.1 Application

Mercury estimation in water is a challenging issue since it is known to be unhealthy for people, and the accumulation of such metal in the human body can significantly damage it. Mercury is usually absorbed by water, air, and soil because of harmful activities like mining or fossil fuel combustion, and its ion (Hg^{2+}) recognition is required in the surface water an early detection task. From this perspective, it is important to develop a compact and inexpensive solution for detecting heavy metal pollution for on-site analysis.

Portable lab-on-hardware is a research field where studies occur for on-site detection or monitoring subtle chemical changes that would have broad applications in organic classification, disease diagnosis, environmental surveillance, or bioresearch. Since smartphones (SP) are lightweight and widely available, they are particularly suited for on-site analysis. Many mobile sensors have been developed for SP using attached 3D printed optical interface with physical detection instruments[139]. For example,[144] proposed an SP-based Mercury detection platform, with an integrated optomechanical attachment to generate contamination maps where GPS coordinates are recorded for each tested water sample. However, this approach may not be ideal since the easiness and portability of an SP are jeopardized by a custom tool. In the same way that an app can be installed on several SPs, a portable solution should be able to work on the fly on any SP device equipped with a camera.

An example of a different approach is described in[122] in which an Android application using a bare SP is able to detect tuberculosis. Overall, SP-based data analysis solutions have shown promising results for analyzing multiple samples and could be used as an alternative to the costly microplate spectrometer reader with the help of Machine Learning (ML) techniques[103].

6.2.2 Automatic recognition

Automatic recognition of analytes in containers, like microplates, is a wide research field, examples of recent extensive literature review can be found in [139] and [4;83]. The literature encompasses different areas related to the problem of recognizing chemical compounds with specific luminescence, thanks to the use of machine learning with smartphone-based data acquisition and data analysis. Smartphones are the most suitable tools for a successful on-site detection for point-of-care testing (POCT) applications [139]. Usually, POCT involves constructing a 3D printed optical interface to extend the smartphone's capabilities. Portable lab-on-hardware is an area in progress since sophisticated imaging apps and technology make point-of-care testing (POCT) applications possible. Smartphone POCT studies occur mainly for on-site surveillance or monitoring testing, especially in remote areas, used for disease diagnosis, environmental monitoring, or bioresearch. This is why also the kind of optical recognition changes, like colorimetric or fluorescence, as well as different compounds like mercury, Measles IgG, secreted antigen from tuberculosis (TB) or glucose. For example, colorimetric detection of Mercury (Hg^{2+}) contamination in water has been done by [60;141;144;145], whereas fluorescence detection was done by [26] and [49].

We describe the literature by taking into account three aspects: first, the relation among acquisition devices and analyte detection; next, computational aspects for the imaging processing part when acquiring photos; finally, statistical/machine learning algorithms that are applied to fit the intensities acquired from the camera and the actual values of the explored compound. Fluorescence or colorimetric biosensors depend on the kind of light source, and image acquisitions can be done by either using only a smartphone or by enriching optical detection with a custom-made extension tool. Optimal images can be mostly acquired when ambient conditions do not change during different photos. Consistency is mostly achieved for photos taken in a closed box housing for smartphones [30;103], or when shooting

conditions are held constant, within a reduced number of photos taken. When such conditions are not held, extensive image processing is required. Biosensors detection or diagnostic devices can generate a wide range of data, processable mostly with regressions techniques, assisted by diverse Machine Learning (ML) solutions[32].

6.3 Microplate Mercury concentration estimate method

This section explains the method used to analyze the microplate and estimate the concentration of Mercury using Supervised Learning algorithms. The section first explains how image data is collected within the different light settings and devices. The images are then processed with Python, and new data features for each well are generated. On such features, two supervised learning algorithms are used: classification for predicting specific classes of concentration and regression for predicting luminescence values. The findings are compared and discussed, focusing attention on how this Proof of Concept (PoC) could potentially be used by natural scientists.

6.3.1 Data Acquisition

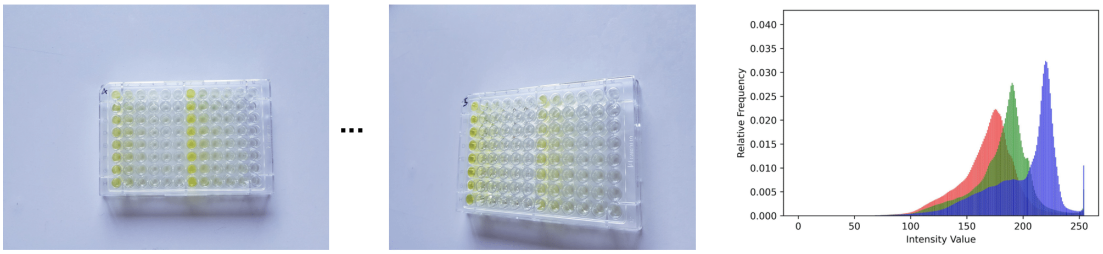
More than 700 mobile shots of a *96-well microplate* (also known as a microtiter plate, MTP, or multiwell) were taken in a laboratory under three different lighting conditions. The 96-well microplate has standard dimensions ($127.71mm \times 85.43mm$) and has 12 columns (1-12) and 8 rows (A-H) for easy well recognition. Photos were taken using the smartphone's automatic exposure setting in three light configurations:

- **ambient light (AB)**, i.e. the light that was already present in the space before any other illumination, basically using the lights in the laboratory, which were made up of artificial lights such as table lamps and neon lights, and natural sunlight from outside;
- **portable UVc (PUVc)**, in which the well plates are directly illuminated in a dark room by a portable UVc emitting lamp placed in the operator's hands, resulting in UV reflection by induced visible fluorescence;
- **UVc**, with a steady UV light coming from behind a fixed well plate, in a dark room.

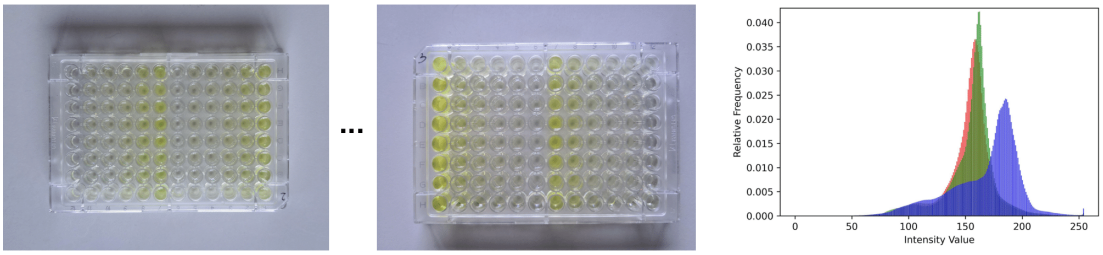
During the tests, two smartphones were used: a Samsung Galaxy S7 and a Huawei Mate 10 Pro. The Samsung G930F Galaxy S7 (model SM-G930F) was launched in March 2016 and featured a 12 MP camera with an F1.7 aperture and a 26mm lens (wide). While the Huawei Mate 10 Pro (model BLA - L09) was released in November 2017 with a 12MP RGB sensor and a 20MP monochrome chip of the Leica Dual camera, equipped with an F1.6 aperture, 27mm (wide), and optical image stabilization. For light settings AB and PUVc, 10 MTPs were prepared, and over 200 images were taken with both the Samsung and Huawei smartphones, photographing each well plate at three different heights and viewpoints with respect to the perpendicular to the plate. For light setting UVc, 20 wells were prepared, and two images were taken for each plate using only the Huawei phone, while holding the smartphone parallel to the surface, centered over the microplate, and at a fixed distance. All of the MTPs (the 10 wellpaltes in scenarios 1 and 2 and the 20 plates of scenario 3) were produced with six concentration classes (100 µM, 10 µM, 100 nmol, 50 nmol, 25 nmol and 0 nmol) and their fluorescence was measured using a spectrometer. Smartphones have been modified for acquiring different luminescence related assays. These assays can be investigated using an excitation light source, but spontaneous light emission should be appropriate for good results[83]. Based on these findings, we decided to extract fluorescent information from Scenario 2.

Figures 6.2, 6.3 and 6.4 illustrates two sample images for each case, each of which depicts one of the three light environments ambient, portable UV, and fixed UV, and in which can be seen two examples of photo shoots with the Mate 10 phone and two with the Galaxy S7 (only for scenario 1 and 2). Furthermore, each Figure contains statistical data, as represented by their respective cumulative histograms, which are discussed in the following paragraphs.

6.3.2 Data Description

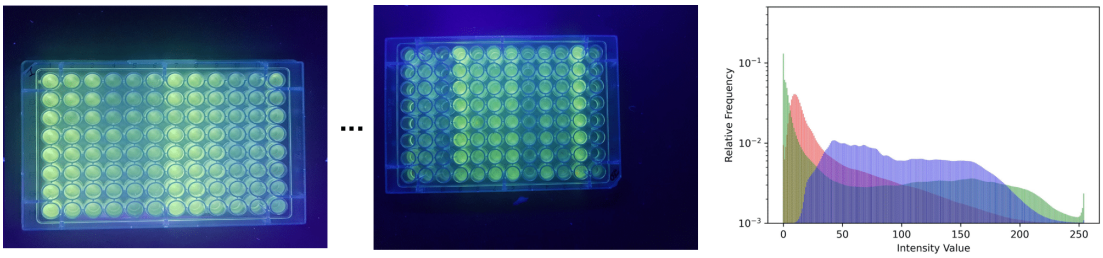


(a) Samsung Galaxy S7, ambient light

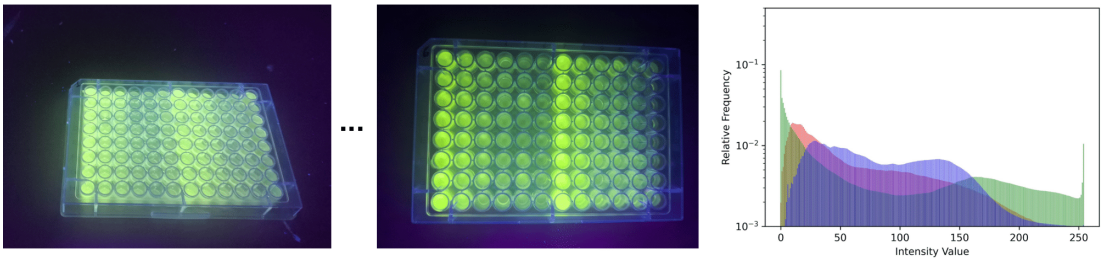


(b) Huawei Mate 10, ambient light

Figure 6.2: For each subfigure, two samples of the 50 photos taken of the multi-well plate are displayed in an ambient light setting, i.e. with a mixture of artificial lights and natural sunlight, and the final figure is the cumulative histogram of all the photos taken in that particular light environment, with that specific phone. Subfigure 6.2b depicts photos taken with the Mate 10 phone, while Subfigure 6.2a depicts photos taken with the Galaxy S7. In 6.2a, one of the chosen samples is mostly perpendicular to the table, while the other is highly tilted, while in 6.2b, both pictures are perpendicular to the table and extremely near to the MTP.



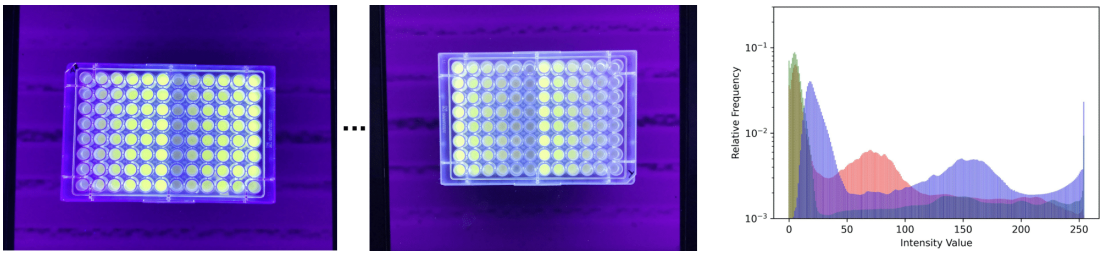
(a) Samsung Galaxy S7, UV-p



(b) Huawei Mate 10, UV-portable

Figure 6.3: Both subfigures display two instances of the photoshoot using a portable UV light, as well as a cumulative histogram. The first two images in each subfigure are examples of the 50 shots taken with each smartphone. The cumulative histogram is represented using a log scale, since the green channel because it is extremely over and underexposed. In 6.3a, one of the examples is captured perpendicular to the late and near, while the other is photographed farther; in 6.3b, the first picture is highly tilted from the bottom perspective, and the second is closed and slightly tilted from the left perspective.

To statistically illustrate the different properties of the photos within the various illumination settings, we grouped all of the photos by camera and setting and measured the sum of all photos, divided by color channel (r, g, b), and the 256 color levels, before normalizing the results. Figures 6.2, 6.3 and 6.4 show the overall histograms of the three light settings used, grouped by smartphone type. Biases in data can be detected by using different lighting as well as different cameras.The overall data can be categorized and analyzed based on the three types of exposures, which combine the types of cameras. As for ambient light, setting, Figure 6.2, there are high levels of RGB, with a predominance of the B channel. Photos taken in both UV settings have mostly green values equal to 0, with minor variations and small values on the other 254 values, making it difficult to note when compared to



(a) Huawei Mate 10, UV

Figure 6.4: Two samples of photos taken in a fixed UV setting. All photos were shot by keeping the smartphone parallel to the table and roughly at the same distance. This testing was investigated using only one smartphone. The last picture is a cumulative histogram in log scale, since the green channel had mostly zero value, and in this graph, it would be difficult to distinguish the data variability unless using a logarithmic scale.

the other two channels. Figures 6.3 and 6.4 show that depending on the type of UV lamp exposure, fixed UV and portable UV have a different predominance of either blue or red. Furthermore, portable UV light produces less distinct patterns. It may be possible to look for a trend by analyzing the mean values with the complete background and specific regions of the well plate. In Section 3.1.6, we will use Machine Learning to uncover such relationships, as well as using LAB color space, which is generally recommended for making quantitative color comparisons for human perception.

6.4 A framework for data extraction and fluorescence prediction

Multiple trail and error development brought to the framework described in Figure 6.5. The framework for evaluating the fluorescence of a well in a microplate can be synthesized in a pipeline composed by the following two main steps:

- a feature extraction procedure: preprocessing and well feature extraction

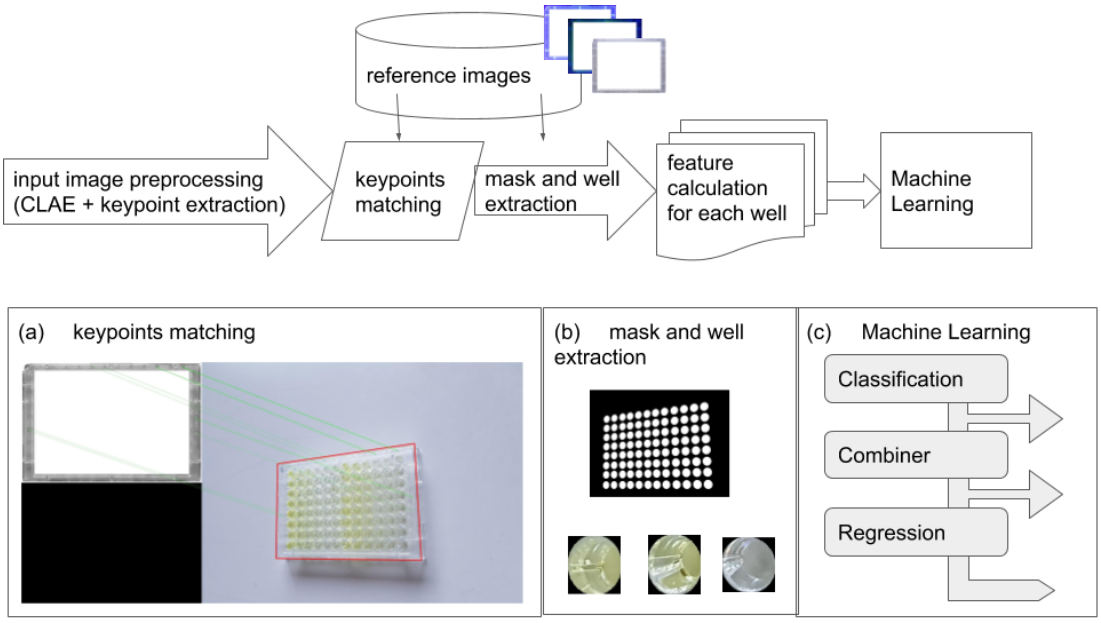


Figure 6.5: Proposed framework. Raw images of well plates are pre-processed and compared to a reference image to determine the location of the well and evaluate the affine transformation. Wells are identified by mapping their locations over the reference image. Multiple features are extracted for each well. These features are fed into machine learning algorithms, which are then trained and tested before being used on new and unknown microplate images.

from microplate, reported in Procedure 1;

- a training procedure: as described in Procedure 2, training and testing of machine learning models (classification and regression).

The pipeline is quite generalized, as certain parameters vary depending on the form of ambient light, but not on the smartphone model. The features are extracted from the input images and used to train a supervised learning model. Since we want to identify the presence of mercury in a well based on its luminescence, quantitative data will be realized by means of color statistics for each well and metadata from the image. Our approach also addresses the issue of non-fluorescent wells appearing to be empty. This explains why it is important to map the location of wells in a reference image before projecting a new image over those images.

We have three types of images in our database: one with natural light and two

with different UV light. A major issue was to identify the structure of the well plate, that was transparent. Color maps have an impact on image data, and it is common practice to project images onto other color maps to gain new insights. As the color transformation matrix varies, various color maps are obtained, and in our tests, we discovered that LAB was the most useful color map in UV setting. As we work with reflected ultraviolet luminescent/fluorescent assays, UVs can highlight boundaries, surface defects, and so reinforce the recognition of the plate well. UVs The configuration of structures is often identified in ultraviolet reflection photography (UVR), as in our case, we work with the reflected-UV imaging rather than the UV fluorescence, since the wellplate structure to identify is around liquid wells. The positions of the well inside the picture were detected by using either b channel or gray. The best channel for reflecting UV is the b channel, while in natural light the well is invisible, the greyscale was the right solution to describe the configuration of the platform by letters and numbers along the edges of the well.

In ambient light, the protein's luminescence in the visible is yellow and we noticed that for selecting as gray, which is an average of the RGB channel, as a single channel we were able to better identify the structure of the wallplate. If the image is captured in a UV setting, given that the fluorescence of the compound is yellow, and since UV generates a predominance of blue color, it is reasonable to choose the b channel of LAB as a unique reference channel to identify the well plate.

6.4.1 Well plate ROI features extraction

The procedure for feature extraction, described in Procedure 1 and represented in Figures 6.6, 6.7 and 6.8 can be summarized with the following steps:

(a) pre-identification of wells on a pre-cropped plate image
(lines 1-2 and Fig. 6.6a- 6.6c);

Procedure 1 Procedure for extracting wells features in a microplate. Single channel for SIFT algorithm can be gray in case of natural light images or can be the b of LAB in UV settings.

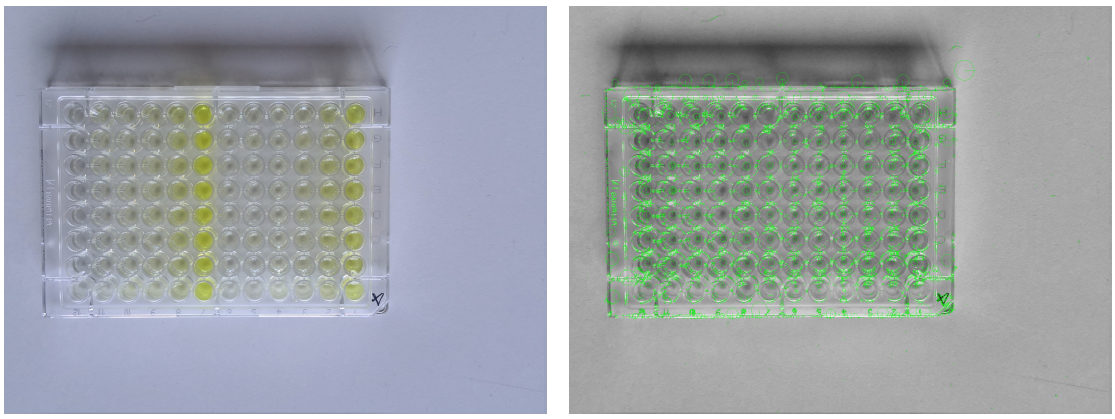
Input: Reference Image Img_R; traing and test set $\mathcal{T}$
apply CLAHE equalization on the Img_R single channel
extract aligned and ordered circle position on the Img_R {using Hough}
extract keypoints and descriptors with SIFT on Img_R border
for each image Img in $\mathcal{T}$ **do**
  crop the biggest rectangle containing the highest gray-level of luminescence within the clustered $k = 2$ on Img
  apply CLAHE equalization on single channel Img
  extract keypoints and descriptors with SIFT on Img
  while $err \geq hqt$ **and** $dr < 0.75$ **do**
    match Img keypoints and descriptors with Img_R
    if $\#matches < min\_matches$ **then**
      increase dr
    else
      calculate transformation matrix homography M and reverse homography R among keypoints with RANSAC
      $err \leftarrow \|I_{3x3} - M \cdot R\|_\infty$
    end if
  end while
  apply a perspective transformation with M to generate a mask by mapping circle position from Img_R
  for each of the 96 wells in mask **do**
    extract all the statistical properties on Img, over the multiple channels
  end for
end for
map wells with true value {in case of training}
Output: Set of features $\mathbf{f}$, with true value v

(a) Reference image (Img_R)

(b) Single channel Img_R, after equalization and keypoints extraction

(c) Position mask of the circles on Img_R after applying Hough algorithm and circle parameters normalization

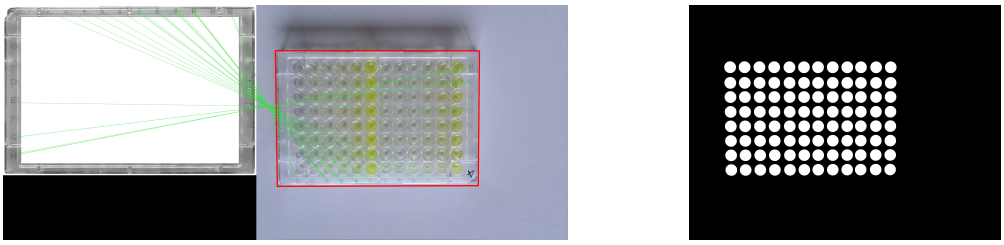
Figure 6.6: Photos of the well extraction method that have been exposed to natural light (presented here for descriptive purposes). These two images show how the reference image (Img_R) was processed for well position extraction.



(a) Example of input Img from camera

(b) Single channel Img, after equalization and keypoints extraction

Figure 6.7: Representative images of the well extraction process using pictures exposed to natural light. The photographs depict the technique used with one of the acquired well-plates, which is upside-down and was acquired slightly angled. The former are images obtained from the smarthopne, whilst the latter are single-channel images with keypoints.



(a) Mapping Img_R with Img with homography

(b) Reverse homography to find the position of the wells in Img

Figure 6.8: Mapping of the well plate of Fig. 6.7, over the reference image of Fig. 6.6, by obtaining the position of the wells in the upside-down image.

(b) MTP identification (lines 6-7) and matching with reference image (lines 8-12);

(c) MTP rotation matrix calculation, with homography (lines 13-14);

(d) ROI (well) identification from the input MTP with feature extraction for each well (lines 17-20).

We fix the positions of the well on a reference image and use its borders as key-points. New well images are then matched on such key-points, and new wells have automatically mapped the potions of the wells according to a transformation. Eventually from the area of each well, numerical features are extracted, ready to be passed to a supervised learning algorithm.

(a) Image prepossessing pre-identification of wells

On each image a process of Contrast limited adaptive histogram equalization (CLAHE) is applied, as in lines 2 and 6, to obtain a single-channel image so to be passed as input of the scale-invariant feature transformation (SFIT) algorithm. Hough circle detection algorithm[153] is applied on the reference image to extract

the positions of the well circle in the well plate image (Figures 6.6, 6.7 and 6.8). Circle detection, with Hough circle detection algorithm[153], gave quite good results in identifying correctly the wells, but, the parameter tweaking needed most analysis and since they were ovals it was not easy to apply to photos that were not perfectly parallel to the desk. By modifying Hough parameters we stop the procedure when the number of circles identified is 96 and their positions, more or less overlap the wells we see on the image. Finally, the values for the center positions and radius are standardized, either by row and column, or by recalculating their sizes with a trimmed mean and thereby recalculating the positions of the circle parameters. The circles will better overlap the wells this way, this finally generates Figure 6.6c. Furthermore, the key point extraction pipeline takes a single channel as input. The standard method for obtaining a single channel is to average RGB values, resulting in a gray image. In the UV setting, since the light emitted is in the ultraviolet (blue) region, the b coordinate (encompassing yellow and blu) resulted in more feasibility to distinguish well plate borders.

(b) Keypoints Extraction and Matching

Our framework uses the SIFT algorithm proposed by[75;76]. The SIFT algorithm identifies and describes keypoints, i.e.points in a picture that are interesting or stand out. Each detected keypoint has a descriptor that is associated with it, and these descriptors are invariant to affine transformation or distortion.SIFT generates is a set of key point-descriptor, (p, s, r, f), where $p = (x, y)$ is the location of the key point pixel on the image, s the scale, r the orientation, and f a 128-dim descriptor generated from local image gradients. Example of generated keypoints are shown in Figures 6.6b and 6.7b, generated, respectively at line 3 and 7 in the Procedure 1. As for lines 9-12, descriptors matching is applied between the border of the reference image and the input image. Once descriptors are built, they need to be compared to find suitable pairs, i.e. a correspondence, in order to match the same objects of the

two photos. In[87], it is implemented the hierarchical k-means tree or randomized kd-trees, and these algorithms formed the Fast Library for Approximate Nearest Neighbors (FLANN), in an OpenCV library.

As we can not reliably decide which match is right, to prevent false matches, we refuse the match. At this stage of the framework it is not possible to decide which match is right, so to prevent false matches, Lowe's solution[17;75] is to reject ambiguous match, using a distance ratio test, the Nearest Neighbor Distance Ratio (NNDR) check, in order to sort the matches and try to avoid fake matches. The NNDR compares the two best matches, if they are very close, it excludes them as an uncertain match. The top two matches should be distinct, and by considering the best match by the second-best match, the NNDR excludes matches greater than a threshold, a distance ratio dr, closest neighbor distance ratio is defined as follows:

$$NNDR = \frac{d_1}{d_2} < dr,$$

where the nearest and second nearest neighbor distances are $d1$ and $d2$.

In our framework, we decided to dynamically choose the best dr (distance ratio) with small increments in the value from 0.68, since the smaller dr is, the better the match. dr is best to be between 0.68 and 0.75, this is why we select the lowers the possible value, and increase up until the matches are qualitatively good.Best matches were achieved when using channel L or b of the LAB color spaces and when cropping the reference image so to leave the border of the well plate alone. An example of a good matching are the green lines in Figure 6.8a. In our implementation, we needed to make sure that the single channel images on with we were calculating the keypoints, were the same as the reference image and the input image. To do so we tested multiple single channels that could give us the best keypoints that represented the well plate. First, to increase the number of matches, we tested different channel configuration, and channel L or b of the LAB color spaces resulted in the best solution to increase the number keypoints

and matches found. In the case of the UV setting, since the light emitted is in the ultraviolet (blue) area, the b coordinate (encompassing yellow and blue) proved to be more suitable for distinguishing well plate borders.

(c) Homography with RANSAC

Since the matching results may return inaccurate correspondences (outliers) as well as correct matches (inliers), at this step of the framework we want to find the relative orientation with a homography while bewaring of outliers. To manage outlier rejection, the homography is calculated with RANSAC (RANdom SAmple Consensus)[44]. RANSAC is a general fitting algorithm that samples the points, then calculates the relative orientations on the sampled data, scores the remaining points, and counts how many inliers or outliers would be with respect to those points; this process is repeated, and the model with the highest score is kept. Managing the number of times the RANSACs should sample to find the best model, given that the distance ratio dr generated enough correct matches, brings to the overall framework a lot of variabilities, and the developing effort was done to automatize this process. The process has been automated within a while cycle in lines 8-16 of Procedure 1, so that some of the key parameters can be modified, and the product of the homography matrix with its inverse can be minimized. This made our architecture more stable without the need to manually check for specific parameters.

Within the Procedure 1 the main settings required to fix three parameters.

$$err = \|I_{3x3} - M \cdot R\|_{\infty},$$

where M is the transformation matrix homography, and M the reverse homography, while I_{3x3} the 3×3 identity matrix and $\|\|_{\infty}$ is the maximum absolute row sum of the difference matrix. The process stops when the product of the homography

matrix with its inverse is less than a threshold. To assess for a reasonable error, we fix an *homography quality threshold*, *hqt*, at 0.7; ideally, this number would be close to zero and gets bigger than 1 when the transformation is deformed. Since we suppose that the operator of the camera, will be photographing a well palate at the best angle possible, this parameter assures that while mapping keypoints they are distributed similarly in the reference image. An example of a good homography is the red rectangle in Figure 6.8a. Once the matching is valid, since we already have already assured that well were correctly mapped in the matching image, we can easily extract ROI's with the reverse homography of the reference mask, as in Figure 6.8b that maps the position of the circles in Figure 6.6c.

(d) Extraction of features from each well

We also decided to add image EXIF (EXchangeable Image file Format) data tag to improve the classification task[42;136]. For each well, we extracted multiple features, the mean and truncated mean at 30%, standard deviation, skewness, and entropy for each channel color, consisting in gray, RGB, LAB, and HSV. in this way, each well had more than 50 features, plus 12 feature of properties coming from the metadata of the phone camera. We extract EXIF data from photos taken by a mobile phone camera, selecting the tags more commonly available in most consumer camera-smartphone, i.e. ShutterSpeedValue, ApertureValue, Brightness-Value, ExposureBiasValue, MaxApertureValue, Flash, FocalLength, ExposureTime, Orientation, FocalLengthIn35mmFilm, FNumber and MeteringMode. At the end of the feature extraction precede, we obtain more than 50 features, multiplied by the 96 well of the plate for each well plate. In the case of a training process, we know what class the well belongs to and what luminescence value it has. This information is then applied to the final data collection, ready to be transferred to the supervised learning algorithms. After that, all of the features are transferred to our combiner for classification and regression.

6.4.2 Supervised learning module

All the wellplates were coupled with the luminosity values of the well, so we were able to use ML algorithms.

The machine learning approach used is supervised learning, composed by the following points:

- classification model fitting or evaluation;
- predicted probability average;
- regression model fitting or evaluation;
- regression values average.

Either for regressing and classification we tested random forest (RF) and Multi-layer Perceptron (MLP) algorithms from[99] and XGBoost (XGB) from[27]. Since empirically we recognized in the dataset, being composed of multiple similar features, we decided to use two tree-like algorithms, one based on begging (RF) and one based on boosting (XGB), plus a neural network (MLP) that is also able to be used on multiple kinds of data. While the final desired result was the ability to calculate the intensity value, and thus the actual concentration of mercury, we used a classification algorithm rather than a regression algorithm. Using a classification algorithm was expected to help us be data-driven, since the content of each well was generated based on a given concentration, but there was some variability in luminescence within the same concentration. As a result, all wells with the same concentration belonged to the same class, and this came to be the first test to validate our assumption in this Proof of Concept (PoC). Classification assesses the adherence of a data point, into a class, by also specifying the probability that it can be part of each of the available classes. The classes available in the experiment setting were six (100 µM, 10 µM, 100 nmol, 50 nmol, 25 nmol and 0 nmol). and these were used ad target class for three classification methods: random forest (RF), boosted trees (XGB), and Multi-layer Perceptron (MLP) algorithm. We

then merged the outcomes of these methods by taking the mean of the likelihood and choosing the class with the highest probability. This method attempted to improve regression by incorporating classification details as new features into the data passed in as input to the regression model.

Let us look into detail at the three models: MLP, RF, and XGB.The Multi-layer Perceptron (MLP) is a feedforward neural network able to solve nonlinear problems, composed of multiple perceptrons disposed in multiple layers. Our implementation had 4 hidden layers, of decreasing size, each time halved, from 2^8 down to 2^5. As an activation function, we used the rectified linear activation function (ReLU), used in many neural networks making the network easier to train and achieve better performances. The random forest, introduced by Breiman, is essentially a bagged decision where sub-sampling is applied to decision trees at each split, by averaging the results of the tree. In the implemented framework, RF uses 50 estimators (i.e. trees). For each tree, not all the feature vector is used, but only a sub-sample, in our implementation we used as the log base 2 of the input size. in our case, with more than 50 features, each tree has a sample of 5 features. Bagging is a typical ensemble technique known also as bootstrap aggregating. Even if it was designed to be applied to classification is used also for Regression problems.And as for the regressor RF, we used the mean absolute error (MAE) criteria for measuring the quality of the tree split.

Ensemble strategy

Ensemble strategies try to use a group of weak learners to produce a strong learner who performs better than a single one. Apart from Bagging another similar technique is Boosting. Boosting learns new models sequentially, in an adaptive way, while Bagging fits multiple models independently and merges their predictions. In such a way, boosting will be less biased. In our framework, we used Extreme Gradient Boosting known as XGBoost (XGB) that is an implementation of gradient

boosting trees algorithm, where XGB fits trees iteratively by decreasing a loss function. In the proposed framework, after fitting the tree models (MLP, RF, and XGB) we also combined their results by using ensemble learning.

Combining supervised models is a known procedure that is used to make results more robust[116]. Ensemble methods weigh several individual models, and combine them in order to improve predictive performance, by using the principles of the wisdom of the crowd. In our framework, we used Weighted Average Probabilities for classification combining rather than majority voting by calculating the class label as argmax of the sum of predicted probabilities. In the case of regression, the combination of different regressors is returned as the average of expected values. In general, given n categories and m classifiers; the (weighted) majority vote for the classifier is defined by:

$$\hat{y} = \arg\max_i \sum_{j=1}^{m} w_j p_{ij}$$

where p_{ij} is the probability assigned to the i-th category by the j-th classifier. In this way we aggregate, i.e. combine, probabilities to generate one final prediction. Similar formula is used for regression

$$\hat{r} = \sum_{j=1}^{m} w_j r_j,$$

where r_j can be considered the value assigned to the i-th category by the j-th classifier. In our case we used $m = 3$ ML models (MLP, RF, and XGB) with $n = 6$ classes for classification, and we fixed $w_j = \frac{1}{3}$ for all j.

Another approach we used to improve the regression results was to combine classification and regression.Connecting the classification output with the regression input can use information from the scientific setting about how the concentrations were produced to train the ML model. Researchers who interpolate unknown concentrations with previously identified concentrations use a similar method. In

line 8, we concatenate the probabilities of each predicted class to the feature vector, when passed in input to the regressor. By using well know ML models (MLP, RF, and XGB) and implementing two techniques (model combination and feature extension) we expect to have enough variability in ML to assess at best the response of ML models while changing dataset.

Procedure 2 Procedure for training and executing classification and regression models and their combination.

Input: Set of vector features **f**
apply MinMax scaler on **f**
if classification **then**
apply MLPClassifier on **f**
apply RandomForestClassifier on **f**
apply XGBClassifier on **f**
calculate for each class mean value of predicted probabilities values **p**
set as output the calss of the combiner the max predicted probability
extend **f** with **p** {so to use, eventually, as input for regression}
end if
if regression **then**
apply MLPRegressor on **f**
apply RandomForestRegressor on **f**
apply XGBRegressor on **f**
calculate mean value of predicted values
end if
Output: Predicted class and/or value for each well of a plate

6.5 Mercury estimation results

This final section reports the results of the developed framework applied to the three experimental settings. We trained the supervised models, by splitting the dataset of wells features into train and test with an 80-20 proportion. The results have been validated with the overall accuracy for the calcification task and with

Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) for the regression part. The Feature extraction was run once for each experimental setting, by fixing the correct reference image, and color channel on which to extract feature keypoints and descriptors for matching. Afterward for each light setting the supervised learning procedure was run 15 times, 5 times for each of the available techniques (classification, regression, and regression with classification features) in order to obtain the average of the evaluation metric over 5 runs. All the result values are reported in Table 6.1 and Table 6.2.

Type	**Exposure**	**Method**	**Average Accuracy**
Only Classification	UV	Combined	**0.96 ± 0.01**
Only Classification	UV	RF	0.94 ± 0.01
Only Classification	UV	XGB	**0.96 ± 0.01**
Only Classification	UV	MLP	0.95 ± 0.02
Only Classification	UV-portable	Combined	**0.85 ± 0.02**
Only Classification	UV-portable	RF	0.79 ± 0.04
Only Classification	UV-portable	XGB	0.81 ± 0.02
Only Classification	UV-portable	MLP	*0.84 ± 0.03*
Only Classification	environment light	Combined	**0.80 ± 0.03**
Only Classification	environment light	RF	0.78 ± 0.03
Only Classification	environment light	XGB	**0.80 ± 0.02**
Only Classification	environment light	MLP	0.74 ± 0.01

Table 6.1: Results Classification evaluated with averaged accuracy. The higher the value for accuracy the better. Averaged value and standard deviation are calculated by repeating 5 times a Holdout validation, that is a 5-Shuffle Split Cross-validation.

As for classification, we report the accuracy, that is the sum of true positives and true negatives divided by the total number of samples, i.e.

$$\texttt{accuracy}(y, y^*) = \frac{1}{|T|} \sum_{(y_i^*, y_i) \in T} \mathbf{B}(y_i^* == y_i)$$

where $|T|$ is the cardinally of the test set, y_i^* is the predicted class of the i-th sample and y_i is the corresponding true class and $\mathbf{B}$ is the Boolean-valued function,

and == is the logical operator.

And then we repeat this calculation 5 times each time shuffling randomly the train and test; Table 6.1 reports the obtained values for the train-test procedure.

As can be seen from Table 6.1, the best accuracy is obtained by the combined classifier in UV setting (96%), and making XBG equally accurate for UV setting and natural light setting, while for portable UV setting the second-best result is with MLP. In general, the combiner obtains the best result in UV, with 96% accuracy, in the portable UV setting with 85% accuracy, and in the natural light setting, with an 80% accuracy. Whereas XGB tends to perform better than RF, underlining the aspect that the trained model of XGB is less biased than RF. It can be seen that our PoC architecture reaches accuracy values a high as 85 or 95 percentage in UV setting and the main difference in the 10 percentage difference is due to the quality of photo acquisition based on the ideal photo condition with a specific camera. The accuracy predicted is quite reassuring. There is high accuracy as 0.95 as the UV in steady-state and as low as 0.85 with a portable UV light on by using multiple cameras. This delta is due to multiple factors. The high inclination in some photos of testing scenarios 1 and 2, leads to a quite reduced number of pixels useful for significant data. The ambient light, as well as portable UV, are quite sensitive to light reflections, and wells luminescence tend to unrelated with the concentration.

Table 6.2, reports the results with regression and regression-with-calcification methodologies. Regression has been evaluated by calculating, the Mean Absolute Error (MAE), the expected value of the l_1-norm loss, and the Root Mean Squared Error (RMSE), which is the expected value of the quadratic error. The Mean absolute error regression loss, estimated over the size of the test set $|T|$, is calculated as:

$$\text{MAE}(y, y^*) = \frac{1}{|T|} \sum_{(y_i, y_i^*) \in T} |y_i - y_i^*|,$$

where y_i^* is the predicted value of the i-th sample, and y_i is the corresponding true

Type	**Exposure**	**Method**	**Average MAE**	**Average RMSE**
Class-Regression	UV	Combined	*1.0 ± 0.1*	**2.7 ± 0.3**
Class-Regression	UV	RF	1.2 ± 0.1	**2.7 ± 0.3**
Class-Regression	UV	XGB	**0.9 ± 0.1**	3.2 ± 0.1
Class-Regression	UV	MLP	1.3 ± 0.1	**2.7 ± 0.1**
Only Regression	UV	Combined	1.8 ± 0.2	3.0 ± 0.1
Class-Regression	UV-portable	Combined	**4.0 ± 0.4**	7.5 ± 0.7
Class-Regression	UV-portable	RF	4.7 ± 0.4	7.7 ± 0.7
Class-Regression	UV-portable	XGB	**4.0 ± 0.5**	9.0 ± 0.7
Class-Regression	UV-portable	MLP	4.3 ± 0.3	**7.3 ± 0.7**
Only Regression	UV-portable	Combined	5.0 ± 0.2	*7.4 ± 0.4*
Class-Regression	environment light	Combined	**3.8 ± 0.6**	*6.7 ± 0.8*
Class-Regression	environment light	RF	4.0 ± 0.3	6.8 ± 0.5
Class-Regression	environment light	XGB	**3.8 ± 0.5**	7.7 ± 1.0
Class-Regression	environment light	MLP	4.1 ± 0.6	**6.6 ± 0.9**
Only Regression	environment light	Combined	4.7 ± 0.1	7.4 ± 0.6

Table 6.2: Results for regression. The lower values for MAE or RMSE, the better. Averaged value and standard deviation are calculated by calculating 5 times either MAE and RMSE with an Holdout validation, i.e. a 5-Shuffle Split Cross validation.

value; while, with the same notation, the root mean square error is calculated as

$$\mathrm{RMSE}(y, y^*) = \sqrt{\frac{1}{|T|} \sum_{(y_i, y_i^*) \in T} \|y_i - y_i^*\|^2}.$$

Also in this case we present the Average and standard deviation of these values, because of the randomness in choosing the 80-20 proportion for training and testing. As can be seen from Table 6.2, within the same light settings, the combined methodology performs better or is the second-best solution with respect to other approaches. Moreover applying only regression (even if is combined) performs worst than any other method that used in input further features extracted from classification. In general, among the three experimental settings, the best results are obtained, within the third setting, of UV lighting. Then among the other two settings, the ambient light environment setting performs better than UV-portable setting.

Final validation was done on a further well plate, with four unknown wells, whose concentrations were unknown to both data scientists and chemists. The same well was acquired under the portable UV-b and in natural lighting, and two ML models trained with their two light settings, give similar results, within an acceptable range. And Natural light settings proved to be more useful for everyday application and better results, according to domain experts. In both regression and classification tasks, the best results are obtained by the combiner, with good performance obtained by XGB. The overall setting (inclination, use of a single camera, and homogeneous illumination condition) make the models perform better, with less variance, and with less difference among ML models.

Chapter 7

Conclusion

Unsupervised data analysis requires concepts of metric and similarity functions for the partitioning of the data into clusters containing "homogeneous" clouds of information. The main idea is to apply unsupervised learning methodologies on such data to perform correlation analysis, also useful to understand and find patterns in data, and obtained clusters. Learning techniques can be also used to focus on bridging the data gap between data and the application decision. In the context of situation awareness, such data are needed to make informed decisions.

This thesis applies the Data Science paradigm to two main areas in the context of Smart Cities: cultural heritage domain and e-health. To begin, this thesis examines unsupervised learning methods for extracting multimedia features from a cultural heritage dataset and a mixed dataset of numerical and categorical values. The second pillar of smart cities, e-health, is addressed through two applications: one for extending a legacy SQL database with Knowledge Graph insights and data management[1], and another as a proof of concept framework for interacting with biosensors using common portable devices such as a smartphone [2].

[1]All of the code pertaining to this KG embedding of medical booking data can be found at `https://github.com/MthBr/kg_embedding_for_medical_booking_data`.

[2]All of the source code pertaining to this PoC can be found at `https://github.com/MthBr/well-plate-light-driven-predictions`.

7.1 Unsupervised learning in the Cultural Heritage case study

We present a study of unsupervised learning techniques applied on IoT data to support decision making processes inside intelligent environments. To assess the proposed approach we discuss two case-of-study in which behavioural IoT data have been collected, also in a non-invasive way, in order to achieve an unsupervised classification that can be adopted during a decision making process. The use of Unsupervised Learning techniques is acquiring a key role to complement the more traditional services with new decision making ones supporting the needs of companies, stakeholders and consumers. Considering the two proposed case-of-study, in both cultural locations, thanks to the adopted unsupervised techniques an increase of less involved visitors from the August month to October month can be observed.

In particular, in the first case-of-study (the M.A.N.N.) users enjoyed the museum without any digital device and showed different behaviours according to their length-of-stay. One of the objective is to infer knowledge from a Cultural Heritage case study represented by the National Archaeological Museum of Naples, where data have been collected by an Internet of Things (IoT) framework. An additional outlook depends on the time spent in the museum entrance and in the main hall only (this behaviour can depend on ex-temporary exhibitions or on people who did not enter the museum for the visit). To cluster the collected data several issues have been addressed. Firstly, we have investigated the selection of the number k of clusters to partition data. Afterwards data analysis by k-medoid and hierarchical clustering algorithms has been discussed. Comparisons between several similarity and distance functions suggest us useful hidden path in the data. In the future works we will consider to extract additional features from our dataset (i.e. the time feature) in order to improve the classification approach.

Looking at the second case-of-study (the Maschio Angioino) the focus of our study was only on the subset of people who entered the museum and wanted to rent and use a multimedia interactive system supporting the visit. Time dimensionality appears there, but also a new dimension depending on the device, i.e. number of clicks, is introduced, that leads to a different partition of data revealing new behaviors. Overall, the time feature seems to have a strong relevance in clustering visitor's behaviours. A preliminary multiemdia feature extraction process has been performed from many log files representing the visitors' behaviour using multimedia interactive devices. Within industrial applications, for museum directors, and cultural heritage stakeholders, that need to make informed decisions. Novel frameworks are required to tackle such problems, in our case, we started with multimedia features in order to analyze visitors' behaviours.

An in-depth analysis of the available dataset, through two case-of-study, has showed hidden behaviours and context-aware situations. Discovering classes of users from unstructured data coming from the Internet of Things framework requires Machine Learning techniques in order (i) to partition a given dataset into a number of clusters and (ii) to support real time decisions and context-aware situations useful for the management of intelligent environments.

7.2 About e-health and KG

KG is an emerging technology able to build large knowledge bases, structured collections of facts relating to a state of the world. It allows you to form a network of linked data belonging to a domain, which constitutes the starting context, which is also connected to other external datasets in a context of increasingly extended relationships. Our aim in this paper has been to introduce a KG approach for the booking systems of the healthcare domain by also exploiting ML methodologies. By extracting medical booking data from the public healthcare system in Campania,

Italy, a KG composed of "patient", "health-service provision", "practitioner", and "booking-staff" has been designed. Starting from an in-depth analysis of this kind of data, generated from the last six years of medical prescriptions and appointment bookings, a KG approach for extracting insights and useful knowledge for stakeholders and healthcare organization is presented and discussed. Exploiting the Graph Embedding technique we have applied on the embedded data both unsupervised (clustering) and supervised learning (classification) approaches to extract hidden knowledge. The experimental tests on a real-world dataset are promising and also highlight that much further research can be carried out. Within the proposed framework, innovative relation prediction (i.e. link prediction) can be performed to assess the task of Knowledge Graph Completion. The classification results that can assess new connections between patients and health service might be the starting point for link prediction. Moreover improved embedding procedures that take into account meta-paths should improve the quality of prediction accuracy. Future works might examine RotatE and ConvE by comparing them it with ComplEx.

Still, there are numerous tasks that must be completed. Investigate various embeddings, particularly nonlinear ones, and try to figure out how to improve them given a specific schema structure. The KGLIB (Knowledge Graph Library) based on Deep Mind Graph Nets as an alternative to embedding approaches. Grakn was especially beneficial for: simply building KG schemas, doing Reasoning, extracting triples, and finally, Grakn can execute Link analysis either through embedding or directly on a graph.

AI is a term these days, but it delivers a lot of wonderful ideas and aims, so it may be worth investing in it to take use of all the data and algorithms accessible. Any data can be improved, including legacy data with noise, however great effort is required in purification and knowledge engineering. It is not an easy task; it needs a significant amount of work, human hours, and high hardware

specifications. There are numerous tools available, including R, Python, and Grakn. However, they must be used correctly, which necessitates a thorough understanding of Machine Learning methodologies as well as an awareness of statistical biases. Producing insights can be challenging due to the aforementioned difficulties. The output of machine learning algorithms must be critically evaluated and thoroughly questioned.

7.3 Mercury estimation

Finally, this work provides a Proof of Concept (PoC) for developing a Point of Diagnosis (PoD) by studying the luminescence of a compound interacting aquired with a smartphone.

The possibilities for such an application are numerous, like measuring the concentration of mercury in water directly on a boat. This approach is feasible since the reaction of a compound with a reacting luminescent agent can be determined, in general, providing support for extensive training data.

Extensions to the Android application are possible, and training on several light natural light settings or possibly the use of a homogeneous dark camera to improve performance are options for improving the tool's accuracy.

Future studies would include the possibility of improving overall classification accuracy and reducing regression errors using various light settings, camera phones, and other types of well plates. A calibration curve should be determined for the final consumer. The first technique to investigate is to enhance ROI extraction by distinguishing between the portion of the liquid and the reflection of light on liquid, as well as the reflection of liquid on the well borders.

Further research may be conducted to increase the performance of the keypoint matching, using various algorithms, such as CNN approaches, or by better using SWIFT, such as looking for keypoints across several channels and then merging

the best matches in the final homography. Color balancing studies can be used to improve image homogeneity, especially when using the gray world assumption and applying similar tweaks to ambient light and UV light settings photographs.

As for the supervised learning enabling methodology, there are three implemented models in this version; further work should add more models to improve accuracy; additionally, there are no specific hyperparameter settings in this version; most stetting were selected empirically or left to default settings, implying that further improvement can be done with automatic hyperparameter optimization to improve accuracy. Furthermore, the combined approach employs an average of values; further research may look at weighted averages. Finally, the current framework can be then implemented in Android, making it accessible by natural scientists.

In summary, this thesis also provides a Proof of Concept (PoC) for developing a Point of Diagnosis, of analyzing the luminescence of reacting compounds with a smartphone, and the preliminary results show the prospects to potentially help chemists test luminescence with their smartphone, using Machine Learning without complicated objects to expand the optics of their camera.

Bibliography

[1] Learning entity and relation embeddings for knowledge resolution. *Procedia Computer Science*, 108:345 – 354, 2017. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2017.05.045. International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland.

[2] *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II*, volume 11779 of *Lecture Notes in Computer Science*, 2019. Springer. ISBN 978-3-030-30795-0. doi: 10.1007/978-3-030-30796-7.

[3] K. J. AbuHassan, N. M. Bakhori, N. Kusnin, U. Z. M. Azmi, M. H. Tania, B. A. Evans, N. A. Yusof, and M. A. Hossain. Automatic diagnosis of tuberculosis disease based on Plasmonic ELISA and color-based image classification. In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 4512–4515, July 2017. doi: 10.1109/EMBC.2017.8037859.

[4] T. Alawsi and Z. Al-Bawi. A review of smartphone point-of-care adapter design. *Engineering Reports*, 1(2):e12039, Sept. 2019. doi: 10.1002/eng2.12039.

[5] M. Ali, S. Vahdati, S. Singh, S. Dasgupta, and J. Lehmann. Improving

access to science for social good. In *ECML-PKDD*, 4th Workshop on Data Science for Social Good, 08 2019.

[6] M. Aria and C. Cuccurullo. bibliometrix: An r-tool for comprehensive science mapping analysis. *Journal of Informetrics*, 11(4):959–975, 2017. URL https://doi.org/10.1016/j.joi.2017.08.007.

[7] M. Ayoub. A review on machine learning algorithms to predict daylighting inside buildings. *Solar Energy*, 202:249–275, May 2020. ISSN 0038-092X. doi: 10.1016/j.solener.2020.03.104. URL https://www.sciencedirect.com/science/article/pii/S0038092X20303509.

[8] C. Badii, P. Bellini, D. Cenni, A. Difino, P. Nesi, and M. Paolucci. Analysis and assessment of a knowledge based smart city architecture providing service APIs. *Future Generation Computer Systems*, 75:14–29, Oct. 2017. ISSN 0167-739X. doi: 10.1016/j.future.2017.05.001. URL https://www.sciencedirect.com/science/article/pii/S0167739X17302273.

[9] G. Bakal, P. Talari, E. V. Kakani, and R. Kavuluru. Exploiting semantic patterns over biomedical knowledge graphs for predicting treatment and causative relations. *Journal of Biomedical Informatics*, 82:189 – 199, 2018. ISSN 1532-0464.

[10] L. Bellomarini, G. Gottlob, A. Pieris, and E. Sallinger. Swift logic for big data and knowledge graphs. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2–10, 2017. doi: 10.24963/ijcai.2017/1. URL https://doi.org/10.24963/ijcai.2017/1.

[11] L. Bellomarini, D. Fakhoury, G. Gottlob, and E. Sallinger. Knowledge graphs and enterprise ai: The promise of an enabling technology. In *2019 IEEE*

35th International Conference on Data Engineering (ICDE), pages 26–37, April 2019. doi: 10.1109/ICDE.2019.00011.

[12] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305, Feb. 2012. ISSN 1532-4435.

[13] A. Berquand, F. Murdaca, A. Riccardi, T. Soares, S. Generé, N. Brauer, and K. Kumar. Artificial intelligence for the early design phases of space missions. In *2019 IEEE Aerospace Conference*, pages 1–20, 2019. doi: 10.1109/AERO.2019.8742082.

[14] C. M. Bishop. Pattern recognition. *Machine learning*, 128(9), 2006.

[15] F. Bohnert, I. Zukerman, S. Berkovsky, T. Baldwin, and L. Sonenberg. Using interest and transition models to predict visitor locations in museums. *AI Commun.*, 21(2-3):195–202, Apr. 2008. ISSN 0921-7126. URL `http://dl.acm.org/citation.cfm?id=1460172.1460183`.

[16] S. Boriah, V. Chandola, and V. Kumar. Similarity Measures for Categorical Data: A Comparative Evaluation. In *SDM*, Proceedings, pages 243–254. Society for Industrial and Applied Mathematics, Apr. 2008. ISBN 9780898716542. doi: 10.1137/1.9781611972788.22. URL `https://epubs.siam.org/doi/abs/10.1137/1.9781611972788.22`.

[17] M. Brown and D. Lowe. Invariant Features from Interest Point Groups. In *In British Machine Vision Conference*, pages 656–665, 2002.

[18] J. Brownlee. A Tour of Machine Learning Algorithms, Aug. 2019. URL `https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/`.

[19] W. Budiaji. *kmed: Distance-Based k-Medoids*, 2019. URL `https://CRAN.R-project.org/package=kmed`. R package version 0.2.0.

[20] H. Cai, V. W. Zheng, and K. Chang. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge & Data Engineering*, 30(09):1616–1637, sep 2018. ISSN 1558-2191. doi: 10.1109/TKDE.2018.2807452.

[21] T. J. Callahan, I. J. Tripodi, H. Pielke-Lombardo, and L. E. Hunter. Knowledge-based biomedical data science. *Annual Review of Biomedical Data Science*, 3(1):23–41, 2020. doi: 10.1146/annurev-biodatasci-010820-091627.

[22] J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, June 1986. ISSN 0162-8828. doi: 10.1109/TPAMI.1986.4767851.

[23] G. Casolla, S. Cuomo, V. S. d. Cola, and F. Piccialli. Exploring Unsupervised Learning Techniques for the Internet of Things. *IEEE Transactions on Industrial Informatics*, 16(4):2621–2628, Apr. 2020. ISSN 1941-0050. doi: 10.1109/TII.2019.2941142.

[24] S.-H. Cha. Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307, 2007.

[25] M. Charrad, N. Ghazzali, V. Boiteau, and A. Niknafs. NbClust: An R package for determining the relevant number of clusters in a data set. *Journal of Statistical Software*, 61(6):1–36, 2014. URL `http://www.jstatsoft.org/v61/i06/`.

[26] B. Chen, J. Ma, T. Yang, L. Chen, P. F. Gao, and C. Z. Huang. A portable RGB sensing gadget for sensitive detection of Hg2+ using cysteamine-capped QDs as fluorescence probe. *Biosensors and Bioelectronics*, 98:36–40, Dec. 2017. ISSN 0956-5663. doi: 10.1016/j.bios.2017.05.032.

[27] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785.

[28] V. Christophides, V. Efthymiou, T. Palpanas, G. Papadakis, and K. Stefanidis. An overview of end-to-end entity resolution for big data. *ACM Comput. Surv.*, 53(6), dec 2020. ISSN 0360-0300. doi: 10.1145/3418896. URL https://doi.org/10.1145/3418896.

[29] L. Ciabattoni, G. Foresi, A. Monteriù, L. Pepa, D. P. Pagnotta, L. Spalazzi, and F. Verdini. Real time indoor localization integrating a model based pedestrian dead reckoning on smartphone and ble beacons. *Journal of Ambient Intelligence and Humanized Computing*, 10(1):1–12, Jan 2019. ISSN 1868-5145.

[30] B. Coleman, C. Coarsey, and W. Asghar. Cell phone based colorimetric analysis for point-of-care settings. *Analyst*, 144(6):1935–1947, Mar. 2019. ISSN 1364-5528. doi: 10.1039/C8AN02521E.

[31] L. Costabello, S. Pai, C. L. Van, R. McGrath, N. McCarthy, and P. Tabacof. AmpliGraph: a Library for Representation Learning on Knowledge Graphs, Mar. 2019. URL https://doi.org/10.5281/zenodo.2595043.

[32] F. Cui, Y. Yue, Y. Zhang, Z. Zhang, and H. S. Zhou. Advancing Biosensors with Machine Learning. *ACS Sensors*, 5(11):3346–3364, Nov. 2020. doi: 10.1021/acssensors.0c01424.

[33] S. Cuomo, P. De Michele, and M. Pragliola. A computational scheme to predict dynamics in iot systems by using particle filter. *Concurrency and Computation: Practice and Experience*, 29(11):e4101, 2017.

[34] S. Cuomo, P. De Michele, F. Piccialli, and A. K. Sangaiah. Reproducing dynamics related to an internet of things framework: A numerical and statistical approach. *Journal of Parallel and Distributed Computing*, 118: 359–368, 2018.

[35] E. Curry, S. Hasan, C. Kouroupetroglou, W. Fabritius, U. ul Hassan, and W. Derguech. Internet of things enhanced user experience for smart water and energy management. *IEEE Internet Computing*, 22(1):18–28, 2018. doi: 10.1109/MIC.2018.011581514.

[36] B. Desgraupes. *clusterCrit: Clustering Indices*, 2018. URL `https://CRAN.R-project.org/package=clusterCrit`. R package version 1.2.8.

[37] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel. Convolutional 2D Knowledge Graph Embeddings. 2018. URL `http://arxiv.org/abs/1707.01476`.

[38] L. R. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945. ISSN 00129658, 19399170. URL `http://www.jstor.org/stable/1932409`.

[39] T. R. dos Santos and L. E. Zárate. Categorical data clustering: What similarity measure to recommend? *Expert Systems with Applications*, 42 (3):1247 – 1260, 2015. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2014.09.012. URL `http://www.sciencedirect.com/science/article/pii/S095741741400551X`.

[40] L. Ehrlinger and W. Wöß. Towards a definition of knowledge graphs. In *SEMANTiCS (Posters, Demos, SuCCESS)*, 2016.

[41] P. Eklund, T. Wray, P. Goodall, and A. Lawson. Design, information organisation and the evaluation of the virtual museum of the pacific digital

ecosystem. *Journal of Ambient Intelligence and Humanized Computing*, 3 (4):265–280, Dec 2012.

[42] Y. Fang, H. Zhu, Y. Zeng, K. Ma, and Z. Wang. Perceptual Quality Assessment of Smartphone Photography. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3677–3686, 2020.

[43] S. Finlayson, P. LePendu, and N. Shah. Building the graph of medicine from millions of clinical narratives. *Scientific Data*, 1, 2014.

[44] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981. ISSN 0001-0782. doi: 10.1145/358669.358692.

[45] J. Fletcher. Grakn KGLIB (Knowledge Graph Library), Jan. 2022. URL `https://github.com/vaticle/kglib`. original-date: 2018-09-16T16:46:48Z.

[46] L. Goasduff. 2 Megatrends on the Gartner Hype Cycle for Artificial Intelligence, 2020, Sept. 2020. URL `https://www.gartner.com/smarterwithgartner/2-megatrends-dominate-the-gartner-hype-cycle-for-artificial-int`

[47] L. Goasduff. The 4 Trends That Prevail on the Gartner Hype Cycle for AI, 2021, Sept. 2021. URL `https://www.gartner.com/en/articles/the-4-trends-that-prevail-on-the-gartner-hype-cycle-for-ai-2021`

[48] J. M. Gómez-Pérez, J. Z. Pan, G. Vetere, and H. Wu. Enterprise knowledge graph: An introduction. In J. Z. Pan, G. Vetere, J. M. Gómez-Pérez, and H. Wu, editors, *Exploiting Linked Data and Knowledge Graphs in*

Large Organisations, pages 1–14. Springer, 2017. ISBN 978-3-319-45652-2. doi: 10.1007/978-3-319-45654-6_1. URL https://doi.org/10.1007/978-3-319-45654-6.

[49] D. Hatiboruah, T. Das, N. Chamuah, D. Rabha, B. Talukdar, U. Bora, K. U. Ahamad, and P. Nath. Estimation of trace-mercury concentration in water using a smartphone. *Measurement*, 154:107507, Mar. 2020. ISSN 0263-2241. doi: 10.1016/j.measurement.2020.107507.

[50] K. Hayashi and M. Shimbo. On the Equivalence of Holographic and Complex Embeddings for Link Prediction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 554–559. Association for Computational Linguistics. doi: 10.18653/v1/P17-2088.

[51] F. Hogenboom, F. Frasincar, U. Kaymak, F. de Jong, and E. Caron. A survey of event extraction methods from text for decision support systems. 85:12–22. ISSN 0167-9236. doi: 10.1016/j.dss.2016.02.006.

[52] A. Ibba, R. Duin, and W.-J. Lee. A study on combining sets of differently measured dissimilarities. In *2010 20th International Conference on Pattern Recognition*, pages 3360–3363, 08 2010. doi: 10.1109/ICPR.2010.820.

[53] J. Jetschni and V. G. Meister. Schema engineering for enterprise knowledge graphs: A reflecting survey and case study. In *2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS)*, pages 271–277, Dec 2017. doi: 10.1109/INTELCIS.2017.8260074.

[54] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*,

pages 687–696, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1067. URL https://www.aclweb.org/anthology/P15-1067.

[55] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu. A Survey on Knowledge Graphs: Representation, Acquisition and Applications. URL http://arxiv.org/abs/2002.00388.

[56] J. Jiang, X. Li, C. Zhao, Y. Guan, and Q. Yu. Learning and inference in knowledge-based probabilistic model for medical diagnosis. *Knowledge-Based Systems*, 138:58–68, 2017.

[57] D. Jurafsky and J. H. Martin. Information Extraction. In *Speech and Language Processing*. 3rd edition. URL https://web.stanford.edu/~jurafsky/slp3/.

[58] F. D. Kakhki, S. A. Freeman, and G. A. Mosher. Evaluating machine learning performance in predicting injury severity in agribusiness industries. *Safety Science*, 117:257 – 262, 2019. ISSN 0925-7535.

[59] T. Kanda, M. Shiomi, L. Perrin, T. Nomura, H. Ishiguro, and N. Hagita. Analysis of people trajectories with ubiquitous sensors in a science museum. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4846–4853. IEEE, 2007.

[60] H. E. Kaoutit, P. Estévez, F. C. García, F. Serna, and J. M. García. Sub-ppm quantification of Hg(II) in aqueous media using both the naked eye and digital information from pictures of a colorimetric sensory polymer membrane taken with the digital camera of a conventional mobile phone. *Analytical Methods*, 5(1):54–58, Dec. 2012. ISSN 1759-9679. doi: 10.1039/C2AY26307F.

[61] L. Kaufman and P. J. Rousseeuw. Clustering by means of medoids, 1987.

[62] A. Khan, S. Uddin, and U. Srinivasan. Understanding chronic disease comorbidities from baseline networks: Knowledge discovery utilising administrative healthcare data. In *Proceedings of the Australasian Computer Science Week Multiconference*, ACSW '17, pages 1–9. Association for Computing Machinery. ISBN 978-1-4503-4768-6. doi: 10.1145/3014812.3014871.

[63] V. Kirchberg and M. Tröndle. The museum experience: Mapping the experience of fine art. *Curator: The Museum Journal*, 58(2):169–193, 2015. doi: 10.1111/cura.12106. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cura.12106.

[64] S. Klarman. Knowledge graph representation: Grakn.ai or owl?, Jan. 2017. URL https://medium.com/vaticle/knowledge-graph-representation-grakn-ai-or-owl-506065bd3f24.

[65] T. Kovács, G. Simon, and G. Mezei. Benchmarking graph database backends—what works well with wikidata? *Acta Cybernetica*, 24(1):43–60, May 2019. doi: 10.14232/actacyb.24.1.2019.5. URL http://cyber.bibl.u-szeged.hu/index.php/actcybern/article/view/3988.

[66] D. Krompaß, M. Nickel, and V. Tresp. Querying factorized probabilistic triple databases. In P. Mika, T. Tudorache, A. Bernstein, C. Welty, C. Knoblock, D. Vrandečić, P. Groth, N. Noy, K. Janowicz, and C. Goble, editors, *The Semantic Web – ISWC 2014*, pages 114–129, Cham, 2014. Springer International Publishing. ISBN 978-3-319-11915-1.

[67] V. Labatut and H. Cherifi. Accuracy measures for the comparison of classifiers. *CoRR*, abs/1207.3790, 2012. URL http://arxiv.org/abs/1207.3790.

[68] J. Lamirel, N. Dugué, and P. Cuxac. New efficient clustering quality indexes. In *2016 International Joint Conference on Neural Networks, IJCNN 2016,*

Vancouver, BC, Canada, July 24-29, 2016, pages 3649–3657, 2016. doi: 10.1109/IJCNN.2016.7727669. URL https://doi.org/10.1109/IJCNN.2016.7727669.

[69] Z. Lei, Y. Sun, Y. Nanehkaran, S. Yang, M. Islam, H. Lei, and D. Zhang. A novel data-driven robust framework based on machine learning and knowledge graph for disease classification. *Future Generation Computer Systems*, 102: 534–548, 2020.

[70] B. Lengerich, A. Maas, and C. Potts. Retrofitting distributional embeddings to knowledge graphs with functional relations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2423–2436, Santa Fe, New Mexico, USA, Aug. 2018. Association for Computational Linguistics.

[71] X. Liang, D. Li, M. Songi, A. Madden, Y. Ding, and Y. Bu. Predicting biomedical relationships using the knowledge and graph embedding cascade model. *PLoS ONE*, 14(6), 2019.

[72] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2181–2187. AAAI Press, 2015. ISBN 0-262-51129-0. URL http://dl.acm.org/citation.cfm?id=2886521.2886624.

[73] T. Lindeberg. Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of Applied Statistics*, 21(1-2):225–270, Jan. 1994. ISSN 0266-4763. doi: 10.1080/757582976.

[74] C. X. Ling, J. Huang, and H. Zhang. Auc: A better measure than accuracy in comparing learning algorithms. In Y. Xiang and B. Chaib-draa, editors,

Advances in Artificial Intelligence, pages 329–341, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. ISBN 978-3-540-44886-0.

[75] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004. ISSN 1573-1405. doi: 10.1023/B:VISI.0000029664.99615.94.

[76] D. G. Lowe. Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image, Mar. 2004.

[77] X. Lv, L. Hou, J. Li, and Z. Liu. Differentiating Concepts and Instances for Knowledge Graph Embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1971–1979. Association for Computational Linguistics. doi: 10.18653/v1/D18-1222.

[78] M. D. Lytras, A. Visvizi, and A. Sarirete. Clustering Smart City Services: Perceptions, Expectations, Responses. *Sustainability*, 11(6):1669, Jan. 2019. ISSN 2071-1050. doi: 10.3390/su11061669. URL `https://www.mdpi.com/2071-1050/11/6/1669`.

[79] M. Maechler, P. Rousseeuw, A. Struyf, M. Hubert, and K. Hornik. *cluster: Cluster Analysis Basics and Extensions*, 2019. R package version 2.1.0 — For new features, see the 'Changelog' file (in the package source).

[80] K. M. Malik, M. Krishnamurthy, M. Alobaidi, M. Hussain, F. Alam, and G. Malik. Automated domain-specific healthcare knowledge graph curation framework: Subarachnoid hemorrhage as phenotype. *Expert Systems with Applications*, page 113120, 2019. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2019.113120.

[81] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[82] C. Martella, A. Miraglia, J. Frost, M. Cattani, and M. van Steen. Visualizing, clustering, and predicting the behavior of museum visitors. *Pervasive and Mobile Computing*, 38:430–443, 2017. ISSN 1574-1192. doi: 10.1016/j.pmcj.2016.08.011.

[83] K. E. McCracken and J.-Y. Yoon. Recent approaches for optical smartphone sensing in resource-limited settings: A brief review. *Analytical Methods*, 8 (36):6591–6601, Sept. 2016. ISSN 1759-9679. doi: 10.1039/C6AY01575A.

[84] A. W. Melton. Problems of installation in museums of art, by arthur w. melton. *Parnassus*, 7(6):29–30, 1935. doi: 10.1080/15436314.1935.11467493. URL `https://www.tandfonline.com/doi/abs/10.1080/15436314.1935.11467493`.

[85] T. Mitchell. Machine learning. 1997.

[86] M. Mittal, V. E. Balas, D. J. Hemanth, and R. Kumar. *Data Intensive Computing Applications for Big Data*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 1st edition, 2018. ISBN 1614998132, 9781614998136.

[87] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *In VISAPP International Conference on Computer Vision Theory and Applications*, pages 331–340, 2009. ISBN 978-989-8111-69-2. doi: 10.5220/0001787803310340.

[88] F. Murtagh and P. Legendre. Ward's hierarchical agglomerative clustering method: Which algorithms implement ward's criterion? *Journal of Classification*, 31(3):274–295, Oct 2014. ISSN 1432-1343. doi: 10.1007/s00357-014-9161-z. URL `https://doi.org/10.1007/s00357-014-9161-z`.

[89] L. Mygind and P. Bentsen. Reviewing automated sensor-based visitor tracking

studies: Beyond traditional observational methods? *Visitor Studies*, 20 (2):202–217, 2017. doi: 10.1080/10645578.2017.1404351. URL https://doi.org/10.1080/10645578.2017.1404351.

[90] W. Nelson, M. Zitnik, B. Wang, J. Leskovec, A. Goldenberg, and R. Sharan. To Embed or Not: Network Embedding as a Paradigm in Computational Biology. *Frontiers in Genetics*, 10, 2019. ISSN 1664-8021. URL https://www.frontiersin.org/article/10.3389/fgene.2019.00381.

[91] H. L. Nguyen, D. T. Vu, and J. J. Jung. Knowledge graph fusion for smart systems: A Survey. 61:56–70. ISSN 1566-2535. doi: 10.1016/j.inffus.2020.03.014.

[92] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich. A Review of Relational Machine Learning for Knowledge Graphs. 104(1):11–33, . ISSN 1558-2256. doi: 10.1109/JPROC.2015.2483592.

[93] M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 809–816. Omnipress, . ISBN 978-1-4503-0619-5.

[94] N. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor. Industry-scale knowledge graphs: lessons and challenges. *Communications of the ACM*, 62(8):36–43, July 2019. ISSN 0001-0782. doi: 10.1145/3331166. URL https://doi.org/10.1145/3331166.

[95] G. B. Orgaz, J. J. Jung, and D. Camacho. Social big data: Recent achievements and new challenges. *Information Fusion*, 28:45–59, 2016.

[96] V. Ortega, L. Ruiz, L. Gutierrez, and F. Cervantes. A selection process of graph databases based on business requirements. In J. Mejia, M. Muñoz,

Á. Rocha, and J. A. Calvo-Manzano, editors, *Trends and Applications in Software Engineering*, pages 80–90, Cham, 2020. Springer International Publishing. ISBN 978-3-030-33547-2.

[97] G. Papadakis, D. Skoutas, E. Thanos, and T. Palpanas. Blocking and Filtering Techniques for Entity Resolution: A Survey. 53(2):31:1–31:42. ISSN 0360-0300. doi: 10.1145/3377455.

[98] H. Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2017.

[99] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

[100] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

[101] E. Pekalska and R. Duin. On combining dissimilarity representations. *Multiple Classifier Systems*, pages 359–368, 01 2001.

[102] A. Pennacchio, F. Giampaolo, F. Piccialli, S. Cuomo, E. Notomista, M. Spinelli, A. Amoresano, A. Piscitelli, and P. Giardina. A machine learning-enhanced biosensor for mercury detection based on an hydrophobin chimera. *Biosensors and Bioelectronics*, 196:113696, Jan. 2022. ISSN 0956-5663. doi: 10.1016/j.bios.2021.113696. URL `https://www.sciencedirect.com/science/article/pii/S0956566321007338`.

[103] N. Phadungcharoen, P. Patrojanasophon, P. Opanasopit, T. Ngawhirunpat, A. Chinsriwongkul, and T. Rojanarata. Smartphone-based Ellman's colourimetric methods for the analysis of d-penicillamine formulation and thiolated polymer. *International Journal of Pharmaceutics*, 558:120–127, Mar. 2019. ISSN 0378-5173. doi: 10.1016/j.ijpharm.2018.12.078.

[104] F. Piccialli, S. Cuomo, V. S. d. Cola, and G. Casolla. A machine learning approach for IoT cultural data. *Journal of Ambient Intelligence and Humanized Computing*, Sept. 2019. ISSN 1868-5145. doi: 10.1007/s12652-019-01452-6. URL https://doi.org/10.1007/s12652-019-01452-6.

[105] F. Piccialli, Y. Yoshimura, P. Benedusi, C. Ratti, and S. Cuomo. Lessons learned from longitudinal modeling of mobile-equipped visitors in a complex museum. *Neural Computing and Applications*, Feb 2019. ISSN 1433-3058. doi: 10.1007/s00521-019-04099-8. URL https://doi.org/10.1007/s00521-019-04099-8.

[106] F. Piccialli, G. Casolla, S. Cuomo, F. Giampaolo, and V. S. di Cola. Decision Making in IoT Environment through Unsupervised Learning. *IEEE Intelligent Systems*, 35(1):27–35, Jan. 2020. ISSN 1941-1294. doi: 10.1109/MIS.2019.2944783.

[107] F. Piccialli, G. Casolla, S. Cuomo, F. Giampaolo, E. Prezioso, and V. S. di Cola. Unsupervised learning on multimedia data: a Cultural Heritage case study. *Multimedia Tools and Applications*, 79(45):34429–34442, Dec. 2020. ISSN 1573-7721. doi: 10.1007/s11042-020-08781-1. URL https://doi.org/10.1007/s11042-020-08781-1.

[108] J. Pujara, H. Miao, L. Getoor, and W. Cohen. Knowledge graph identification. In H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira, L. Aroyo, N. Noy, C. Welty, and K. Janowicz, editors, *The Semantic Web*

– *ISWC 2013*, pages 542–557, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[109] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2018. URL `https://www.R-project.org/`.

[110] M. I. Razzak, M. Imran, and G. Xu. Big data analytics for preventive medicine. *Neural Computing and Applications*, Mar 2019. doi: 10.1007/s00521-019-04095-y.

[111] A. P. Reynolds, G. Richards, B. de la Iglesia, and V. J. Rayward-Smith. Clustering rules: A comparison of partitioning and hierarchical clustering algorithms. *Journal of Mathematical Modelling and Algorithms*, 5(4):475–504, Dec 2006. ISSN 1572-9214. doi: 10.1007/s10852-005-9022-1.

[112] I. Robinson, J. Webber, and E. Eifrem. *Graph Databases*. O'Reilly, 2 edition, 2015. ISBN 978-1-4919-3089-2.

[113] B. Robson. Extension of the Quantum Universal Exchange Language to precision medicine and drug lead discovery. Preliminary example studies using the mitochondrial genome. 117:103621. ISSN 0010-4825. doi: 10.1016/j.compbiomed.2020.103621.

[114] M. Rotmensch, Y. Halpern, A. Tlimat, S. Horng, and D. Sontag. Learning a health knowledge graph from electronic medical records. *Scientific Reports*, 7(1), 2017. doi: 10.1038/s41598-017-05778-z.

[115] L. Rumeng, J. Abhyuday N, and Y. Hong. A hybrid Neural Network Model for Joint Prediction of Presence and Period Assertions of Medical Events in Clinical Notes. 2017:1149–1158. ISSN 1942-597X. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5977733/`.

[116] O. Sagi and L. Rokach. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4):e1249, 2018. ISSN 1942-4795. doi: 10.1002/widm.1249.

[117] S. Sarawagi. Information Extraction. 1(3):261–377. ISSN 1931-7883. doi: 10.1561/1900000003.

[118] A. Sari, W. Rahayu, and M. Bhatt. Archetype sub-ontology: Improving constraint-based clinical knowledge model in electronic health records. *Knowledge-Based Systems*, 26:75–85, 2012.

[119] D. Sarkar. *Lattice: Multivariate Data Visualization with R*. Springer, New York, 2008. URL `http://lmdvr.r-forge.r-project.org`. ISBN 978-0-387-75968-5.

[120] D. Schraivogel, T. M. Kuhn, B. Rauscher, M. Rodríguez-Martínez, M. Paulsen, K. Owsley, A. Middlebrook, C. Tischer, B. Ramasz, D. Ordoñez-Rueda, M. Dees, S. Cuylen-Haering, E. Diebold, and L. M. Steinmetz. High-speed fluorescence image-enabled cell sorting. *Science*, 375(6578):315–320, 2022. doi: 10.1126/science.abj3013. URL `https://www.science.org/doi/abs/10.1126/science.abj3013`.

[121] M. G. Seok and D. Park. A novel multi-level evaluation approach for human-coupled iot applications. *Journal of Ambient Intelligence and Humanized Computing*, Jul 2018. ISSN 1868-5145.

[122] A. M. Shabut, M. Hoque Tania, K. T. Lwin, B. A. Evans, N. A. Yusof, K. J. Abu-Hassan, and M. A. Hossain. An intelligent mobile-enabled expert system for tuberculosis disease diagnosis in real time. *Expert Systems with Applications*, 114:65–77, Dec. 2018. ISSN 0957-4174. doi: 10.1016/j.eswa.2018.07.014.

[123] Q. shan Wang, Z. Mao, B. Wang, and L. Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29:2724–2743, 2017.

[124] Y. Shen, L. Zhang, J. Zhang, M. Yang, B. Tang, Y. Li, and K. Lei. Cbn: Constructing a clinical bayesian network based on data from the electronic medical record. *Journal of Biomedical Informatics*, 88:1–10, 2018.

[125] J. Shinavier and R. Wisnesky. Algebraic property graphs, 2019.

[126] A. Smirnova and P. Cudré-Mauroux. Relation Extraction Using Distant Supervision: A Survey. 51(5):106:1–106:35. ISSN 0360-0300. doi: 10.1145/3241741.

[127] P. Sondhi, J. Sun, H. Tong, and C. Zhai. Sympgraph: A framework for mining clinical notes through symptom relation graphs. pages 1167–1175, 2012.

[128] K. Sookhanaphibarn and R. Thawonmas. A movement data analysis and synthesis tool for museum visitors' behaviors. In P. Muneesawang, F. Wu, I. Kumazawa, A. Roeksabutr, M. Liao, and X. Tang, editors, *Advances in Multimedia Information Processing - PCM 2009*, pages 144–154, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-10467-1.

[129] T. Sørensen. *A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content and Its Application to Analyses of the Vegetation on Danish Commons*. Biologiske skrifter. I kommission hos E. Munksgaard, 1948. URL `https://books.google.it/books?id=rpS8GAAACAAJ`.

[130] F. Sparacino. The Museum Wearable: real-time sensor-driven understanding of visitors' interests for personalized visually-augmented museum experi-

ences. In *In: Proceedings of Museums and the Web (MW2002*, pages 17–20, 2002. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.7.5282.

[131] D. Steinley and M. J. Brusco. Initializing K-means Batch Clustering: A Critical Evaluation of Several Techniques. *Journal of Classification*, 24(1): 99–121, June 2007. ISSN 1432-1343. doi: 10.1007/s00357-007-0003-0.

[132] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. URL http://arxiv.org/abs/1902.10197.

[133] P. Suwinski, C. Ong, M. H. T. Ling, Y. M. Poh, A. M. Khan, and H. S. Ong. Advancing personalized medicine through the application of whole exome sequencing and big data analytics. *Frontiers in Genetics*, 10:49, 2019. doi: 10.3389/fgene.2019.00049.

[134] A. S. Syed, D. Sierra-Sosa, A. Kumar, and A. Elmaghraby. IoT in Smart Cities: A Survey of Technologies, Practices and Challenges. *Smart Cities*, 4(2):429–475, June 2021. doi: 10.3390/smartcities4020024. URL https://www.mdpi.com/2624-6511/4/2/24.

[135] S. Ting, S. Kwok, A. Tsang, and W. Lee. A hybrid knowledge-based approach to supporting the medical prescription for general practitioners: Real case in a hong kong medical center. *Knowledge-Based Systems*, 24(3):444–456, 2011.

[136] W. Trenholm, M. Alexiuk, H. DANG, S. MALEKTAJI, and K. DARCHINI-MARAGHEH. System and method for identification and classification of objects, May 2019.

[137] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33rd*

International Conference on International Conference on Machine Learning - Volume 48, ICML'16, pages 2071–2080. JMLR.org, 2016. URL http://dl.acm.org/citation.cfm?id=3045390.3045609.

[138] M. van der Loo. The stringdist package for approximate string matching. *The R Journal*, 6:111–122, 2014. URL https://CRAN.R-project.org/package=stringdist.

[139] S. K. Vashist and J. H. T. Luong. Smartphone-Based Point-of-Care Technologies for Mobile Healthcare. In S. K. Vashist and J. H. Luong, editors, *Point-of-Care Technologies Enabling Next-Generation Healthcare Monitoring and Management*, pages 27–79. Springer International Publishing, Cham, 2019. ISBN 978-3-030-11416-9. doi: 10.1007/978-3-030-11416-9_2.

[140] L. Vendramin, R. J. G. B. Campello, and E. R. Hruschka. Relative clustering validity criteria: A comparative overview. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 3(4):209–235, 2010. doi: 10.1002/sam.10080.

[141] L. Wang, B. Li, F. Xu, X. Shi, D. Feng, D. Wei, Y. Li, Y. Feng, Y. Wang, D. Jia, and Y. Zhou. High-yield synthesis of strong photoluminescent N-doped carbon nanodots derived from hydrosoluble chitosan for mercury ion sensing via smartphone APP. *Biosensors and Bioelectronics*, 79:1–8, May 2016. ISSN 0956-5663. doi: 10.1016/j.bios.2015.11.085.

[142] Q. Wang, P. Huang, H. Wang, S. Dai, W. Jiang, J. Liu, Y. Lyu, Y. Zhu, and H. Wu. CoKE: Contextualized Knowledge Graph Embedding. URL http://arxiv.org/abs/1911.02168.

[143] R. Wang, M. Wang, J. Liu, M. Cochez, and S. Decker. Structured query construction via knowledge graph embedding. *Knowledge and Information*

Systems, 62(5):1819–1846, Sept. 2019. doi: 10.1007/s10115-019-01401-x. URL https://doi.org/10.1007/s10115-019-01401-x.

[144] Q. Wei, R. Nagi, K. Sadeghi, S. Feng, E. Yan, S. J. Ki, R. Caire, D. Tseng, and A. Ozcan. Detection and Spatial Mapping of Mercury Contamination in Water Samples Using a Smart-Phone. *ACS Nano*, 8(2):1121–1129, Feb. 2014. ISSN 1936-0851. doi: 10.1021/nn406571t.

[145] W. Xiao, M. Xiao, Q. Fu, S. Yu, H. Shen, H. Bian, and Y. Tang. A Portable Smart-Phone Readout Device for the Detection of Mercury Contamination Based on an Aptamer-Assay Nanosensor. *Sensors*, 16(11):1871, Nov. 2016. doi: 10.3390/s16111871.

[146] C. Xie, B. Yu, Z. Zeng, Y. Yang, and Q. Liu. Multilayer Internet-of-Things Middleware Based on Knowledge Graph. *IEEE Internet of Things Journal*, 8 (4):2635–2648, Feb. 2021. ISSN 2327-4662. doi: 10.1109/JIOT.2020.3019707.

[147] W. Xiong, T. Hoang, and W. Y. Wang. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. URL http://arxiv.org/abs/1707.06690.

[148] Z. Yan, J. Liu, L. T. Yang, and N. Chawla. Big data fusion in internet of things. *Information Fusion*, 40:32 – 33, 2018. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2017.04.005.

[149] Y. Yoshimura, S. Sobolevsky, C. Ratti, F. Girardin, J. P. Carrascal, J. Blat, and R. Sinatra. An analysis of visitors' behavior in the louvre museum: A study using bluetooth data. *Environment and Planning B: Planning and Design*, 41(6):1113–1131, 2014. doi: 10.1068/b130047p. URL https://doi.org/10.1068/b130047p.

[150] Y. Yoshimura, A. Krebs, and C. Ratti. Noninvasive bluetooth monitoring

of visitors' length of stay at the louvre. *IEEE Pervasive Computing*, 16(2): 26–34, April 2017. ISSN 1536-1268. doi: 10.1109/MPRV.2017.33.

[151] M. Yu, G. Li, D. Deng, and J. Feng. String similarity search and join: A survey. 10(3):399–417, . ISSN 2095-2236. doi: 10.1007/s11704-015-5900-5.

[152] S. Y. Yu, S. R. Chhetri, A. Canedo, P. Goyal, and M. A. A. Faruque. Pykg2vec: A Python Library for Knowledge Graph Embedding. . URL http://arxiv.org/abs/1906.04239.

[153] H. Yuen, J. Princen, J. Illingworth, and J. Kittler. Comparative study of Hough Transform methods for circle finding. *Image and Vision Computing*, 8(1):71–77, Feb. 1990. ISSN 0262-8856. doi: 10.1016/0262-8856(90)90059-E.

[154] M. Zancanaro, T. Kuflik, Z. Boger, D. Goren-Bar, and D. Goldwasser. Analyzing museum visitors' behavior patterns. In C. Conati, K. McCoy, and G. Paliouras, editors, *User Modeling 2007*, pages 238–246, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-73078-1.

[155] L. Zhan and X. Jiang. Survey on Event Extraction Technology in Information Extraction Research Area. In *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pages 2121–2126. doi: 10.1109/ITNEC.2019.8729158.

[156] S. Zhang, Y. Tay, L. Yao, and Q. Liu. Quaternion Knowledge Graph Embeddings. . URL http://arxiv.org/abs/1904.10281.

[157] Y. Zhang, Q. Yao, Y. Shao, and L. Chen. NSCaching: Simple and Efficient Negative Sampling for Knowledge Graph Embedding. . URL http://arxiv.org/abs/1812.06410.

[158] Z. Zhu, S. Xu, M. Qu, and J. Tang. GraphVite: A High-Performance

CPU-GPU Hybrid System for Node Embedding. In *The World Wide Web Conference*, pages 2494–2504. ACM.

[159] M. Zitnik, F. Nguyen, B. Wang, J. Leskovec, A. Goldenberg, and M. M. Hoffman. Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities. *Information Fusion*, 50:71 – 91, 2019. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2018.09.012.