FAKULTÄT FÜR
INFORMATIK

Otto-von-Guericke-University Magdeburg

Faculty of Computer Science

Department of Databases and Software Engineering

Representation Learning on Electronic Health Records using Graph Neural Networks

Master Thesis

Author:

Altaf Mohammed Aftab

Examiner and Supervisor:

Prof. Dr. rer. nat. habil. Gunter Saake

Supervisor:

Jun.-Prof.Dr. Robert Heyer

(Faculty of Technology, Bielefeld University, Multidimensional Omics Analyses group, Leibniz-Institut für Analytische Wissenschaften – ISAS – e.V.)

Dr.-Ing. David Broneske

(German Center for Higher Education Research and Science Studies (DZHW), Database and Software Engineering Group, Otto von Guericke University)

M.Sc. Daniel Walke

(Database and Software Engineering Group, Otto von Guericke University)

M.Sc. Daniel Micheel

(Database and Software Engineering Group, Otto von Guericke University)

Magdeburg, 01.02.2023

Altaf Mohammed Aftab
Representation Learning on Electronic Health Records using Graph Neural Networks
Master Thesis, Otto-von-Guericke-University Magdeburg, 2023.

Abstract

The digitization of healthcare has led to a proliferation of electronic health records, providing valuable data for machine learning algorithms(e.g., graph neural networks) to make accurate predictions about patients' outcomes(e.g., mortality prediction). This thesis investigates different aspects of representation learning on electronic health records using a graph neural network and provides new insights into the area of graph modelling, feature ablation, and helps to understand the effect of bias in the graph structure. It also evaluates the model's underlying predictors with well-established statistical models (SAPS-II & SAPS-III) for predicting the mortality of the patients diagnosed with sepsis using the MIMIC-III dataset. The experimentation shows that the lab and vital signs features used in predicting mortality in SAPS-II and SAPS-III are ranked in the top 90 percentile amongst the predictors of mortality in the used heterogeneous Graph Attention Network (GAT) model. Experimentation with different graph representations(different ways of representing data in nodes and edges in a heterogeneous graph) shows their advantages and disadvantages. However, in terms of area under receiver operating characteristics curve (Area under the receiver operating curves (AUROC)), different representations performed similarly well. The general way of modelling time-dependent measurements with multiple edges without any aggregation or transformation of edge data had no bias but performed worse in GPU utilization and memory usage. Different ways of encoding the categorical and text data also had an impact on the model's performance, wherein the encoding of such data with a clinical text pre-trained UMLSBert model had better performance than the label or one-hot encoding. Furthermore, the GAT model is tested by introducing an additionally highly biased relationship (similar demography). It was seen that the model's attention mechanism corrected such a nature of bias. Finally, the experiments showed that drugs were the best predictors of mortality among labs, vitals, or diagnoses.

keywords: Digitization, healthcare, Electronic health records, machine learning, mortality prediction, graph neural networks, Sepsis

Acknowledgements

I would like to express my sincere gratitude to my supervisors and mentors Jun.-Prof.Dr. Robert Heyer, Dr.-Ing. David Broneske, M.Sc. Daniel Walke & M.Sc. Daniel Micheel for their continuous supervision, motivation, and expert advice, without which it would have been difficult to get my thesis to fruition.

Your expertise and mentorship have been invaluable to me and have made a significant impact on my personal and professional growth.

Your dedication to your students and willingness to go above and beyond to provide guidance and support has truly been an inspiration to me. Your wealth of knowledge and experience have been invaluable resources that I will carry with me for the rest of my life.

Thank you for your unwavering support and encouragement. Your mentorship has been a critical component in my academic and professional journey, and I am grateful for the opportunity to have learned from someone as talented and dedicated as yourself.

Thank you once again for all that you have done for me. I am grateful for the positive impact you have had on my life, and I wish you all the best in your future endeavours.

A special thanks to Dr.-Ing. David Broneske, M.Sc. Bala Gurumurthy and M.Sc. Chukwuka Victor Obionwu for resolving my untimely requests to restart the GPU server.

Statement of Authorship

I hereby declare that I am the sole author of this master thesis and that I have not used any sources other than those listed in the bibliography and identified as references. I further declare that I have not submitted this thesis at any other institution in order to obtain a degree.

Magdeburg, 01.02.2023
Place, Date


Signature

Contents

List of Figures

List of Tables

List of Abbreviations

MIMIC Medical Information Mart for Intensive Care

SOI Severity of Illness

ICU Intensive care unit

G Graph

V Vertex

E Edge

NN Neural Network

ANN Artificial Neural Network

MLP Multi-Layer Perceptron

GNN Graph Neural Network

GCN Graph Convolutional Network

GAT Graph Attention Network

MF Matrix Factorization

SVD Singular Value Decomposition

FC Fully Connected

Conv Convolution

ReLU Rectified Linear Unit

MSE Mean Squared Error

SGD Stochastic Gradient Descent

CPU Central Processing Units

GPU Graphics Processing Units

DNN Deep Neural Networks

SAGE SAmple and aggreGatE

SVM Support Vector Machines

XGBoost Extreme Gradient Boosting

ROC Receiver operating curves

AUROC Area under the receiver operating curves

EHR Electronic health record

RDBMS Relational database management systems

SQL Structured query language

NoSQL Non-relational structured query language

GNN Graph Neural Network

NLP Natural Language Processing

PCA Principal component analysis

CNN Convolutional Neural Network

CI Confidence Interval

1 Introduction

Digitization has a significant impact on the present-day world, transforming many aspects of society and the way we live and work. The healthcare domain is also undergoing transformations due to ongoing digitization. The digitization of medical health records (e.g., diagnoses, medications, notes, procedures, laboratory events, etc.) into Electronic Health Records (Electronic health record (EHR)) is one of the many areas of the healthcare domain that has the potential to revolutionize the healthcare sector. It improves patient care [KKC+15]; provides analysis of disease evolution [PWH15]; embeds performance measures in clinical practices; identifies patients for clinical trials [EJCH05], and assesses new treatments to gauge their success.

Most often EHRs are stored in relational databases, as they are mature and well understood technology[MQX14]. However, many studies ([EL14] [GBR21] [MQX14]) have shown different benefits of using non-relational databases for storing highly connected EHRs. NoSQL stores are a varied family of technologies classified into four primary models or varieties based on their data model: key-value stores, column-value stores, document stores and graph stores [DCL18], [Cat11]. A graph is a type of data structure with nodes (also referred to as "vertices") & edges (also referred to as "links"), wherein these nodes and edges can contain features [W+01]. A graph database can contain single, or multitudes of such graphs [RWE15]. A systematic literature review by [SDKGG20] shows the increasing trend of representing the EHRs in graph structure and further investigates promising areas.

A single interaction of a patient with the healthcare system produces a ton of data with many relationships. The Figure 1.1 illustrates the interaction schema of a patient. Each node in the schema is again associated with features. E.g., a patient node contains age, gender, ethnicity, etc., and each patient can have multiple encounters (e.g. each patient can be connected to multiple admissions in a hospital) with the healthcare system, portraying a highly connected, complex graph structure. Furthermore, graphs have a flexible structure, and thus the relationship with existing nodes can be extended to new nodes [W+01]. For instance, the drug nodes in the EHR graph can be linked to the proteins they target for finding a new drug (for a given protein target) or finding new targets for a given drug or for finding interactions resulting in adverse side effects when taking multiple drugs. Graphs are ubiquitous and have a wide range of applications, not just in bio-medical sciences but also in other domains such as electrical engineering, mathematics, computation networks, neurosciences, and many more[Pir07]. One disadvantage of graphs is that they are inherently complex for the existing machine learning algorithms to exploit the rich information between the nodes as relationships. Current machine learning algorithms work very well with a fixed structure, such as that found in text or images but cannot process graphs. The process of representing or encoding the graph structure so that machine learning algorithms can easily exploit them is called graph representation learning. Graph representation learning has several applications in

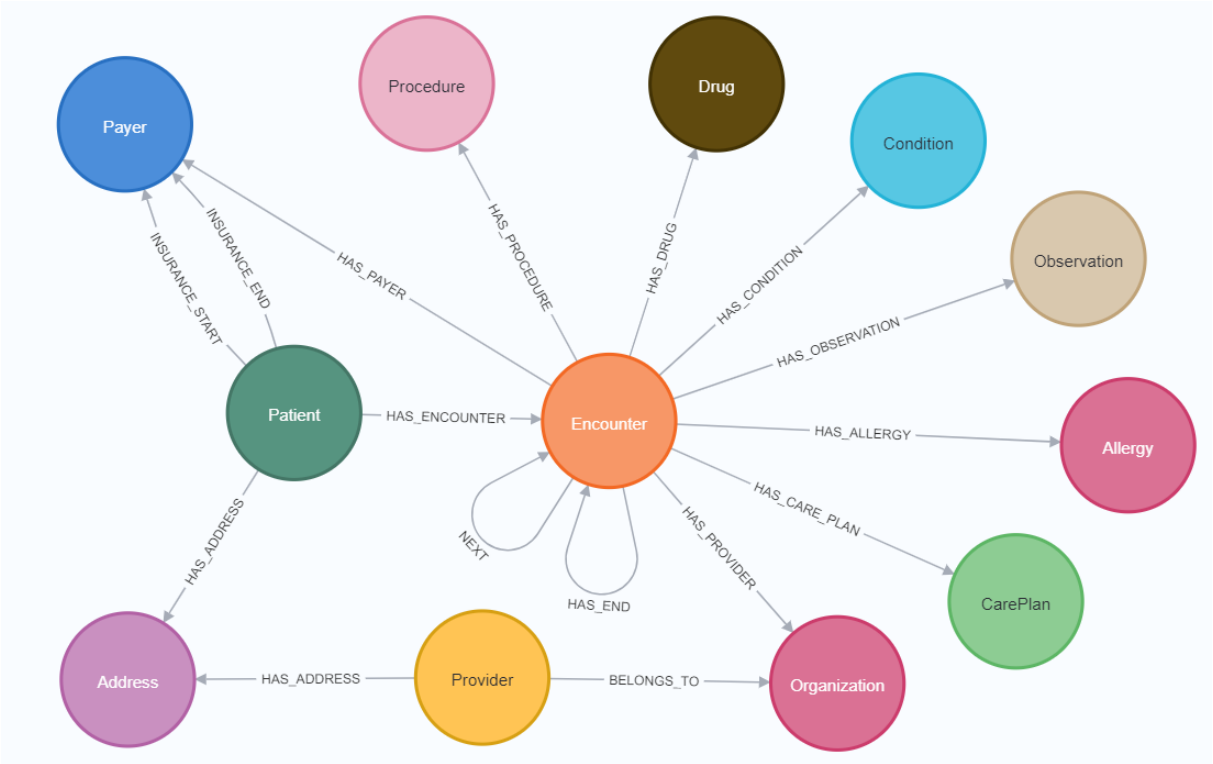


Figure 1.1: Patient & Health Care system Interaction

bio-medicine such as drug discovery, gene function prediction, pathway analysis, cancer research, drug repositioning and medical imaging analysis [YYHK22]. One of such many applications of applying representation learning to EHRs is mortality prediction [LMD15] [CMH+16].

The goal of this thesis is directed towards the mortality prediction of patients who have been diagnosed with sepsis through the use of graph representation learning. Keeping this goal in mind, the subsequent sections are structured as follows:

- Section 1.1 begins with the motivation & reasons behind the goal of the thesis.
- Section 1.2 highlights the main contributions of this research.
- Section 1.3 concludes this chapter by outlining the structure of subsequent sections.

1.1 Motivation

The massive digitization of data in all facets of life generates data at a pace that was never experienced before. The healthcare domain generates an especially large amount of data. As per the report published by Seagate[RRG18] shown in figure 1.2 it estimates a 36% annual growth rate of data in the healthcare domain by 2025, which is the highest amongst the prominent domains.

Digitizing the data has many advantages like easy accessibility, preservation, searchability, disaster recovery, reduced carbon footprint and many more [SFED+12]. The chart 1.3 illustrates the factors influencing the acceptance of digitization in the healthcare domain.

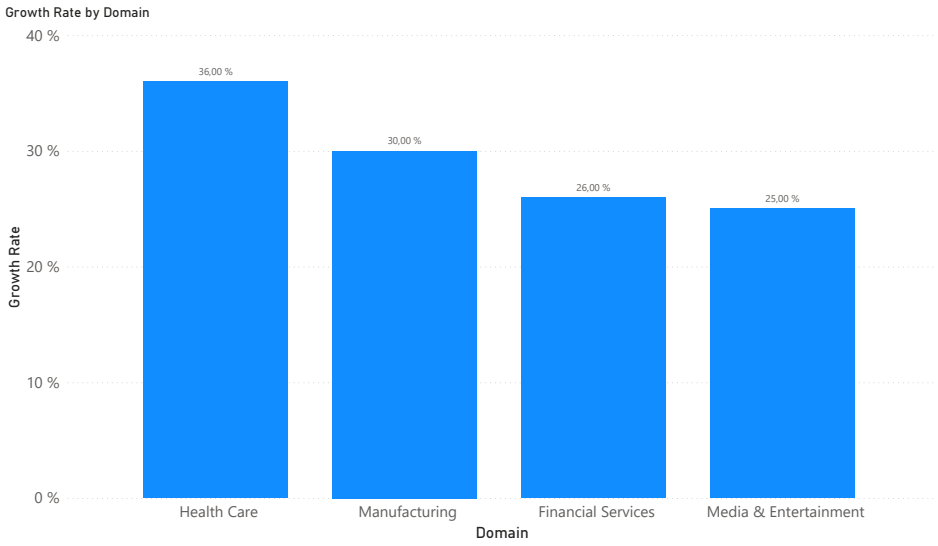


Figure 1.2: Data growth rate in different domains [RRG18]

A McKinsey study along with the German Managed Care Association (BMC)[HBM19] suggests that 26 digital solutions (such as Medical chatbots, Chronic disease management tools, disease prevention tools, Hospital logistics robots, Clinicians' virtual assistants and many more) as shown in figure 1.4 can deliver up to EUR 36 Billion in savings. Although this digitization opens up many potential research areas in the biomedical domain, it also poses serious problems for storing and analysing this complex data. Traditionally, relational databases are used for storing the data because of their popularity. However, with the recent advent of NoSQL stores, the popular trend has started shifting towards this technology, especially for graph databases, which is explained in depth in the background section 2.4. Graph stores are one of the NoSQL stores and their flexible structure makes them versatile. After modelling & storing the data in the graph stores, the next important task becomes analysing it. There are many real-world applications of graphs in the fields of social networks [LC13], computer vision traffic prediction [LYH+20], protein-interaction networks [VFMV03], bioinformatics [LCS+06], the World Wide Web, and several other applications [KW16]. One of many such applications in the domain of health care is mortality prediction [ABEDMB17] [PPC+15]. The idea of mortality prediction is important because it helps

- in assessing the severity of illness
- in adjudicating the value of novel treatments [AAK+21]
- in intervening of health care policies. [FLNG+20]
- in reducing the massive burden of care.
- in enabling the provision of appropriate and timely medical services. [ERESA+20]

This thesis primarily focuses on the mortality prediction 2.2 of patients using graph representation learning on EHRs. However, the concept and methodology introduced in

this work can also be applied to any other targeted medical condition with appropriate pre-processing.

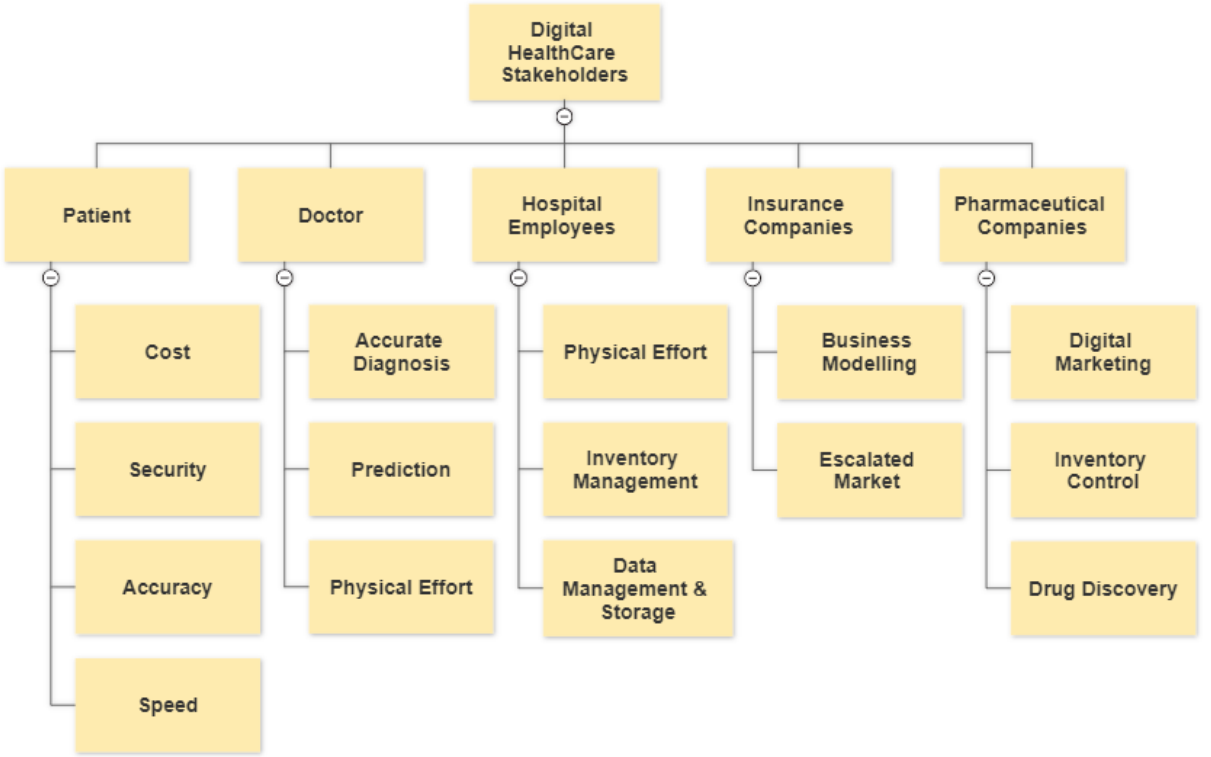


Figure 1.3: Factors influencing acceptance of digitization in healthcare [Alh]

1.2 Main Contributions

The main contributions of this thesis involves investigation of different aspects of representation learning on electronic health records using a graph neural network and provides new insights into the area of graph modelling, feature ablation, and helps to understand the effect of bias in the graph structure. It also evaluates and ranks the model's underlying predictors with that of well established statistical model's predictors, showing that the model learns on significant predictors. The feature ablation experiments on textual data also showed the effect of different encoding techniques on the model.

1.3 Thesis Structure

The rest of the thesis is structured as follows:

- The Chapter 2, gives a deeper understanding of Sepsis, Mortality prediction, graphs & its types, popularity of graph databases, the existing prediction models, representation learning, graph features and state-of-the-art algorithms in graph machine learning which further reinforces & provides clarity on the research goals.

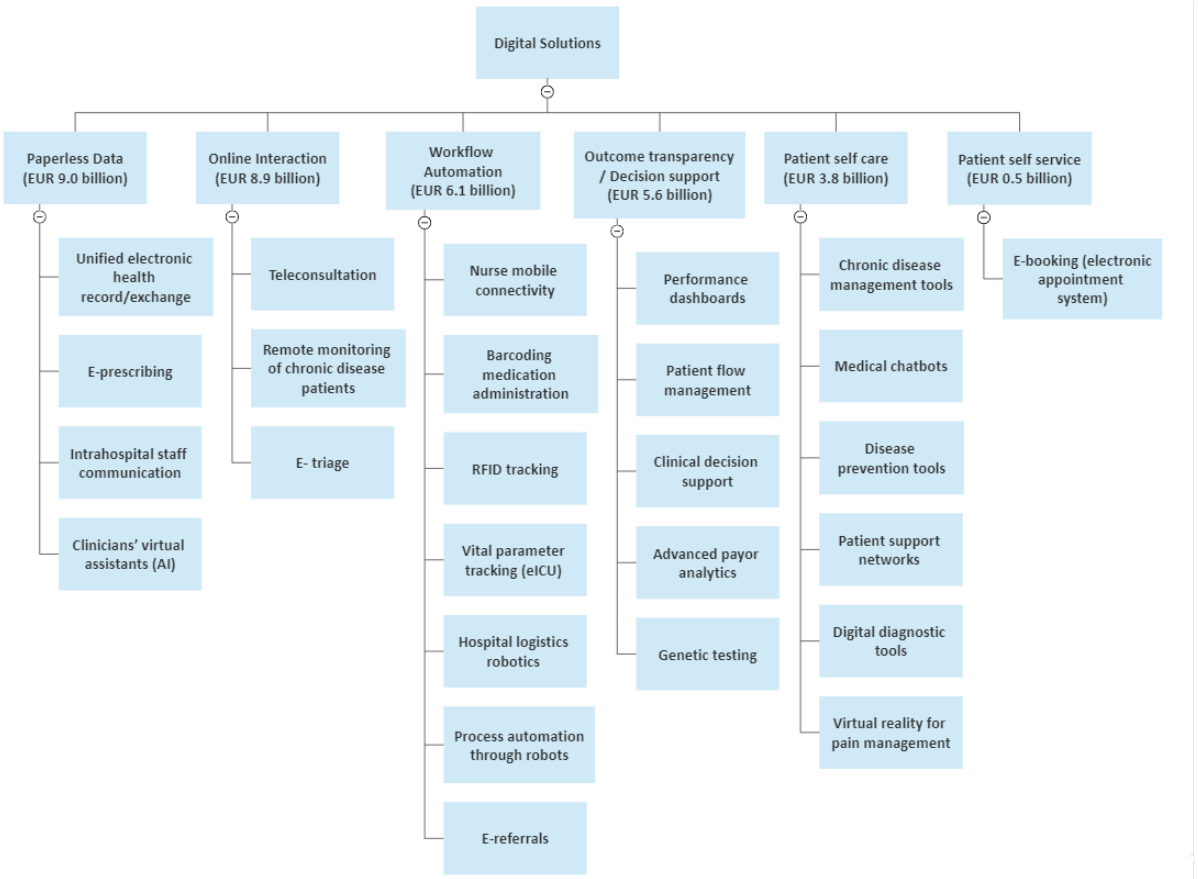


Figure 1.4: Digital solutions [HBM19]

- Chapter 3, reviews existing research in the area of mortality prediction in both traditional machine learning and graph neural network domains.
- Chapter 4, explains briefly about the dataset, an alternative synthetic dataset and provides a complete overview of the database.
- Chapter 5 formulates the goals of this thesis. Specific Research questions are defined, and the design pipeline is explained, including data pre-processing and model implementation.
- In Chapter 6, the essential elements of the experimental setup, such as the graph machine learning model that is used, hyper-parameter of the algorithms, metric used for evaluation, programming frameworks and hardware details that could provide the same results are documented.
- Chapter 7 provides the results of different experiments and evaluates them.
- Finally, Chapter 8 concludes this thesis and outlines some intriguing directions for future research.

2 Background

This chapter presents the background knowledge of this thesis, providing an overview of the essential topics covered in our work. This chapter aims to guide the reader towards a solid foundational understanding of our research area. This chapter is structured as follows:

- In Section 2.1, the focus is on Sepsis as a life-threatening condition, its causes, signs & symptoms, who are at risk and a few statistics about this condition.
- Section 2.2 defines what mortality prediction is, why it is important & what are the existing methodologies in this area.
- In sections 2.3, 2.3.3, 2.5 graph structures, different machine learning tasks on graphs, their challenges are discussed and move on to understand the representation learning on the graphical structure which is the main area of interest in this thesis.
- To predict mortality we use digitized clinical data (EHR). These can be stored in different databases. In the Section 2.4, I introduce the SQL & NoSql database and how graph databases spar against RDBMS, especially in the bio-medical domain.
- In Section 2.7, an in-depth discussion about Graph neural networks (GNN) is given, specifically about Graph convolutional network (GCN), GraphSAGE and Graph attention networks (GAT) as these are most widely used in the graph machine learning domain.

2.1 Sepsis [LM07] [1]

Sepsis or septicemia, or blood poisoning, is a life-threatening condition caused by the body's ferocious reaction to an infection. The onset of Sepsis occurs when an already present infection sets off a cascade of damaging events throughout the body. Sepsis is typically caused by infections in the lungs, urinary tract, skin, or digestive tract. Sepsis is a potentially fatal condition that causes fast tissue damage, organ failure, and death if the infection is left untreated.

2.1.1 What causes Sepsis? [1]

Infections are a forefront cause of this condition. If these infections are left untreated, then they can cause Sepsis. Bacterial infections cause most cases of Sepsis, but viral infections such as COVID-19 or Influenza can cause sepsis.

[1] https://www.cdc.gov/sepsis/what-is-sepsis.html

2.1.2 What are the signs & symptoms of Sepsis? [LM07] [2]

A person with this condition can have one or more of the following conditions:

- High heart rate or weak pulse
- Fever, Shivering or feeling very cold
- Confusion or disorientation
- Shortness of breath
- Extreme pain or discomfort
- Clammy or sweaty skin

Note: A medical assessment by a healthcare professional is required to confirm Sepsis.

2.1.3 Who is at risk? [2]

All are prone to Sepsis, but certain groups are at a higher risk of contraction:

- Adults 65 years or older
- Individuals with weak immunity
- Individuals with an underlying chronic condition such as diabetes, lung disease, cancer and kidney disease.
- Infants.
- Individuals who already survived Sepsis

2.1.4 Statistical facts about Sepsis [2]

According to the CDC (Center for Disease Control & Prevention), in a typical year, at least 1.7 million adults develop Sepsis in the US alone. Of those, 350,000 die during their hospitalization or are discharged to hospice. 1 in 3 patients who die in hospitals in the US has Sepsis. Sepsis, or the infection causing Sepsis, starts before the patient goes to the hospital in about 87% of cases.

[2] https://www.cdc.gov/sepsis/what-is-sepsis.html

2.2 Mortality Prediction

Mortality prediction estimates the probability of an individual's death within a specified time frame. It is often used in healthcare settings to help determine the likelihood of a patient's survival and guide treatment decisions. It is based on various factors, including the patient's demographic data (gender, age, ethnicity, socio-economic status) [PJA+21], medical history, current health status, and the severity and type of the illness or injury they are experiencing. Predictive models may be used to analyze this information and generate a prediction of the patient's likelihood of survival.

Mortality prediction can be helpful for healthcare providers in several ways. For example, it can help providers identify patients at high risk of death which may benefit from more aggressive treatment. It can also help providers identify patients with a better prognosis which may be candidate patients for less intensive treatment. However, estimates of mortality risk are derived from evaluating aggregated data from vast & diverse groups of patients. This means that their validity in the context of each patient encounter cannot be guaranteed. Personal mortality risk estimation, which is addressed in detail in [[LMD15], [LM17]], can help address this deficiency, but this is beyond the scope of the current investigation. It is important to note that the accuracy of predictions depends on the quality of the data used and the complexity of the prediction model. As such, mortality predictions should be used as one factor among many in treatment or decision-making rather than being used in isolation.

There are many different approaches to predicting mortality, and the state-of-the-art models can vary depending on the specific context and goals of the prediction. Some common approaches to mortality prediction include:

- *Statistical models:* These models use statistical techniques, such as regression analysis or survival analysis, to predict the likelihood of death based on various predictors, such as age, gender, medical history, and current health status.
- *Machine learning models:* These models employ algorithms to discover data patterns and generate predictions based on those patterns. They can be trained on large datasets of patient data and can be more accurate than statistical models in some cases.
- *Clinical prediction models:* Clinicians develop these models based on clinical experience and expert judgment rather than statistical analysis. They may include a combination of clinical factors, such as vital signs and lab values, to predict a patient's likelihood of survival.
- *Risk scores:* These models use a specific set of risk factors, such as age, comorbidities, and severity of illness, to assign a score that reflects the patient's risk of death. Higher scores are associated with a higher risk of death.

The performance of each model depends on various factors involved in prediction's unique context and objectives. Before implementing a prediction model in practice, it is essential to verify it to guarantee that it is accurate and dependable.

Severity of Illness (SOI) is a generic measure that provides a patient's discourse from normal physiological behaviour. It categorizes medical conditions as mild, moderate, major,

or critical. This measure offers a framework for analyzing hospital resource usage or setting patient care standards. Several such measures have been implemented in the Intensive care unit (ICU) to forecast different outcomes. These scores are one of the ways of predicting mortality in critically ill patients. There are many different scoring systems used to predict the outcome of critically ill patients, such as the Simplified Acute Physiology Score (SAPS) [LGLS93], the Sequential Organ Failure Assessment (SOFA) [VMT+96], the Mortality Probability Model (MPM) [LTK+93], the APACHE scores [KDWZ85], and many more.

Typically, these models are evaluated using a metric known as Area Under Receiver Operating Characteristics (AUROC), representing the degree of separability for a binary classification. Its value is between 0 and 1, where 1 represents perfect classification between true & false classes and 0 means the complete opposite. Generally, the value is between 0.5-1, where 0.5 means that the model has randomly predicted the true & false classes.

Although AUROC is typically between 0.8-0.9 for the above-discussed models, different approaches, mainly in the area of machine learning & deep learning, are being explored to further improve predictive power by capitalizing on the increased completeness and expressivity of contemporary EHR's. For example, finding and using data from comparable patients at a granular level (i.e., a rich set of clinical variables recorded in high temporal resolution) might lead to constructing a tailored prediction model for any given patient.

2.2.1 SAPS-II

The Simplified Acute Physiology Score version 2 (SAPS II) [LGLS93] is a risk prediction model used to predict critically ill patients' mortality. It is a statistical model that is based on a combination of patient characteristics and medical history.

it is based on a logistic regression model, a statistical method used to predict the probability of a binary outcome (such as mortality). It models the relationship between the predictor variables (such as age, underlying medical conditions, and physiological measurements) and the outcome (mortality) as a logistic curve. It estimates the probability of the outcome for a given set of predictor variables.

It was developed to provide a standardized method for predicting the mortality of critically ill patients, and it has been widely used in intensive care units (ICUs) around the world. It is designed to easily calculate and provide a reliable mortality prediction based on a relatively small number of predictor variables. It is often used with other risk prediction models to provide a more accurate prediction of patient outcomes. The complete breakdown of the scores is explained in the table 2.1 The in-hospital mortality is the calculated as follows In-hospital mortality, $\% = \frac{e^x}{1+e^x}$
where x = 7.7631 + 0.0737 * (SAPS II Score) + 0.9971 * [ln(SAPS II Score + 1)]

Variable	Description	Reference range	Points
Age, years		<40	0
		40-59	7
		60-69	12
		70-74	15
		75-79	16
		80	18
Heart rate	Worst value in 24 hours; if patient has had both cardiac arrest (11 points) and extreme tachycardia (7 points), assign 11 points	<40	11
		40-69	2
		70-119	0
		120-159	4
		160	7
Systolic BP, mm Hg	Worst value in 24 hours	<70	13
		70-99	5
		100-199	0
		200	2
Temperature 39ºC (102.2ºF)	Highest temperature in 24 hours	No	0
		Yes	3
GCS	Lowest value in 24 hours; if patient is sedated, use estimated GCS before sedation	14-15	0
		11-13	5
		9-10	7
		6-8	13
		<6	26
PaO/FiO, if on mechanical ventilation or CPAP	Lowest value in 24 hours; if patient was extubated <24 hours ago, use lowest value while on mechanical ventilation	<100 mm Hg/% (13.3 kPa/%)	11
		100-199 mm Hg/% (13.3-26.5 kPa/%)	9
		200 mm Hg/% (26.6 kPa/%)	6
		Not on mechanical ventilation or CPAP within the last 24 hours	0
BUN, mg/dL (serum urea, mmol/L)	Highest value in 24 hours	BUN <28 or urea <10	0
		BUN 28-83 or urea 10-29.6	6
		BUN 84 or urea 30	10
Urine output, mL/day	If patient in ICU <24 hours, calculate for 24 hours (e.g. if 1 L in 8 hours, then mark 3 L in 24 hours)	<500	11
		500-999	4
		1,000	0
Sodium, mEq/L or mmol/L	Worst value in 24 hours	<125	5
		125-144	0
		145	1
Potassium, mEq/L	Worst value in 24 hours	<3.0	3
		3.0-4.9	0
		5.0	3
Bicarbonate, mEq/L	Lowest value in 24 hours	<15	6
		15-19	3
		20	0
Bilirubin	Highest value in 24 hours	<4.0 mg/dL (<68.4 µmol/L)	0
		4.0-5.9 mg/dL (68.4-102.5 µmol/L)	4
		6.0 mg/dL (102.6 µmol/L)	9
WBC, x 10^3/mm^3	Worst value in 24 hours	<1.0	12
		1.0-19.9	0
		20.0	3
Chronic disease		None	0
		Metastatic cancer	9
		Hematologic malignancy	10
		AIDS	17
Type of admission	Scheduled surgical = surgery scheduled 24 hours in advance	Scheduled surgical	0
	Medical = no surgery within one week of admission	Medical	6
	Unscheduled surgical = surgery scheduled 24 hours in advance	Unscheduled surgical	8

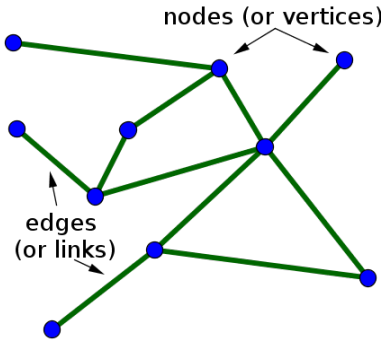
Table 2.1: SAPS II scoring Sheet [LGLS93]

2.2.2 SAPS-III

The Simplified Acute Physiology Score III (SAPS-III) [MMA+05] is a scoring system used to assess disease severity and forecast mortality risk in critically ill patients. It is a weighted scoring system that combines physiological and demographic characteristics to produce scores ranging from 0 to 163. The severity of the patient's sickness and anticipated mortality risk is proportional to the score. The SAPS-III score is derived from a combination of the variables in the table 2.2. The score is widely utilised in ICUs and other critical care settings to assess the severity of illness, forecast mortality risk and monitor the response to treatment. It is a popular and a well-validated instrument for risk assessment and benchmarking patient outcomes in intensive care units. The in-hospital mortality for this score is formulated as In-hospital mortality, $\% = \frac{e^x}{1+e^x}$
where x = 32.6659 + ln(SAPS-3 score + 20.5958) * 7.3068

2.3 Graphs

Graphs are ubiquitous (e.g., transportation network, power grid, supply chain network); the definitions of real-world objects frequently depend on how they link to other entities(e.g., In a transportation network, cities act as entities and they are linked using highways). A graph is a natural representation of a collection of entities and their relationships. Before getting into machine learning on graphs, let us first understand what graphs are. Graphs in mathematics are also called as 'Networks', and graph theory is a branch of mathematics that deals with the study of graphs. Graphs consist of nodes (also referred to as vertices) and edges (also referred to as links), the nodes are connected using the edges. A graph G can be described as $G = (V(x),E(y))$ (V: Nodes, E: Edges, x: Node features, y: Edge features). Figure 2.1 represents the nodes and edges on an undirected example graph.



[Nyk]

Figure 2.1: Example of an Undirected graph

2.3.1 Types of Graphs

Graph can model different characteristics with nodes and edges. Different graphs or networks are utilized to model these properties as shown in Figure 2.2. Graphs can be either un-directed (two-way relationship between two nodes) or directed (one-way relationship

Variable	Description	Reference range	Points
ICU admission	Every patient gets an offset of 16 points for being admitted (to avoid negative SAPS 3 scores)		16
Age, years		<40	0
		40-59	5
		60-69	9
		70-74	13
		75-79	15
		80	18
Comorbidities	Chemotherapy, immunosuppression, radiotherapy, steroid treatment	Cancer therapy	3
		ChronicHF (NYHA IV)	6
		Haematological cancer	6
		Cirrhosis	8
		AIDS	8
		Metastatic cancer	11
Length of stay before ICU admission, day		<14	0
		14-27	6
		28	7
Intrahospital location before ICU admission		Emergency room	5
		Other ICU	7
		Other ward	8
Use of major therapeutic options before ICU admission	Not all variables collected were included in the final data model, please see original article in "Evidence" for further information.	Vasoactive drugs	3
		Other/none	0
Planned or unplanned ICU admission		Planned	0
		Unplanned	3
Reason(s) for ICU admission	If both reasons are present, only the worse value (-4) is scored	Cardiovascular: rhythm disturbances	-5
		Neurologic: seizures	-4
		Cardiovascular: hypovolemic hemorrhagic shock, hypovolemic non-hemorrhagic shock	3
		Digestive: acute, abdomen, other	3
		Neurologic: coma, stupor, obtunded patient, vigilance disturbances, confusion, agitation, delirium	4
		Cardiovascular: septic shock	5
		Cardiovascular: anaphylactic shock, mixed and undefined shock	5
		Hepatic: liver failure	6
		Neurologic: focal neurologic deficit	7
		Digestive: severe pancreatitis	9
		Neurologic: intracranial mass effect	10
		Other	0
Surgical status at ICU admission		Sceduled surgery	0
		No surgery	5
		Emergency surgery	6
Anatomical site of surgery	Not all variables collected were included in the final data model, please see original article in "Evidence" for further information.	Transplant surgery	-11
		Trauma surgery	-8
		Cardiac surgery: CABG without valvular repair	-6
		Other surgery	0
		Neurosurgery: cerebrovascular accident	5
Acute infection at ICU admission	Not all variables collected were included in the final data model, please see original article in "Evidence" for further information.	Nosocomial	4
		Respiratory	5
		Other/none	0
Glasgow Coma Scale/Score	Lowest within 1 hr of ICU admission	3-4	15
		5	10
		6	7.5
		7-12	2
		13	1
Total bilirubin, mg/dL (µmol/L)	Highest within 1 hr of ICU admission	<2 mg/dL (<34.2 µmol/L)	1
		2-5.9 mg/dL (34.2-102.5 µmol/L)	4
		6 mg/dL (102.6 µmol/L)	5
Body temperature, °C (°F)	Highest within 1 hr of ICU admission	<35 °C (<95 °F)	7.5
		35 °C (95 °F)	1
Creatinine, mg/dL (µmol/L)	Highest within 1 hr of ICU admission	(3-4 µmol/L)	15
		(5 µmol/L)	10
		(6 µmol/L)	7.5
		<1.2 mg/dL (<106.1 µmol/L)	1
		1.2-1.9 mg/dL (106.1-176.7 µmol/L)	2
		2-3.4 mg/dL (176.8-309.3 µmol/L)	7
		3.5 mg/dL (309.4 µmol/L)	8
Heart rate, beats/min	Highest within 1 hr of ICU admission	<120	1
		120-159	5
		160	7
Leukocytes, G/L	Highest within 1 hr of ICU admission	<15	1
		15	2
pH	Lowest within 1 hr of ICU admission	7.25	3
		>7.25	1
Platelets, G/L	Lowest within 1 hr of ICU admission	<20	13
		20-49	8
		50-99	5
		100	1
Systolic blood pressure, mm Hg	Lowest within 1 hr of ICU admission	<40	12
		40-69	8
		70-119	3
		120	1
Oxygenation	PaO2, FiO2 refer to arterial oxygen pressure (lowest), Inspiratory oxygen concentration MV	PaO2/FiO2<100 and MV	11
		PaO2/FiO2100 and MV	7
		PaO2<60 and no MV	5
		PaO260 and no MV	1

Table 2.2: SAPS-III Scoring Sheet [MMA+05]

indicated by the direction of the edge). Similarly, weighted graphs can have a weight associated with an edge, indicating the importance between two nodes. A graph consisting of single types of nodes and edges, is homogeneous, and the ones consisting of different node types or different edge types are heterogeneous [WJS+19]. For E.g., in metabolic networks, nodes represent different enzymes, and there exist different enzyme interactions between them. Similarly, in protein-metabolites network there exists different node types (proteins and metabolites) which interact with each other. Understanding different types of graphs in "Graph theory" helps us come up with the choice of representation for a specific domain. Although in some domains, the representation is unique & unambiguous, it could be ambiguous in others. The study's success depends on the choice of graph representation. Since graphs are a natural way of representing information, they have the same underlying schema (i.e nodes, edges, node features and edge features), so a single machine-learning algorithm developed for graph structure should be able to solve any network with some adaption for dealing with homogeneous and heterogeneous graphs.

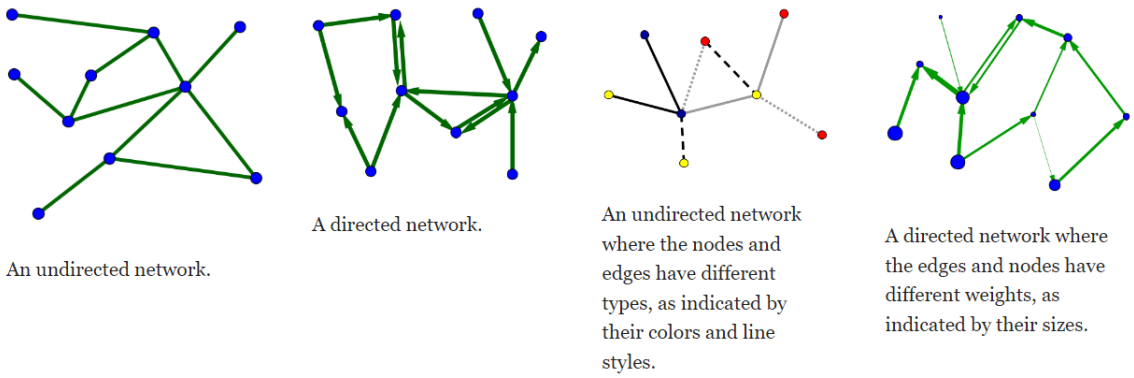


Figure 2.2: Types of Graphs [Nyk]

2.3.2 Types of tasks on graphs

Generally, there are three different types of prediction tasks on graphs, Node-level tasks, Edge-level tasks and Graph-level tasks reference to figure 2.3

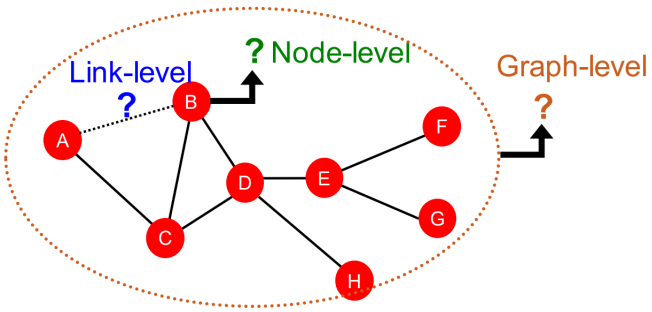


Figure 2.3: Tasks in Graphs [3]

[3]http://web.stanford.edu/class/cs224w/slides/02-tradition-ml.pdf

Node level tasks

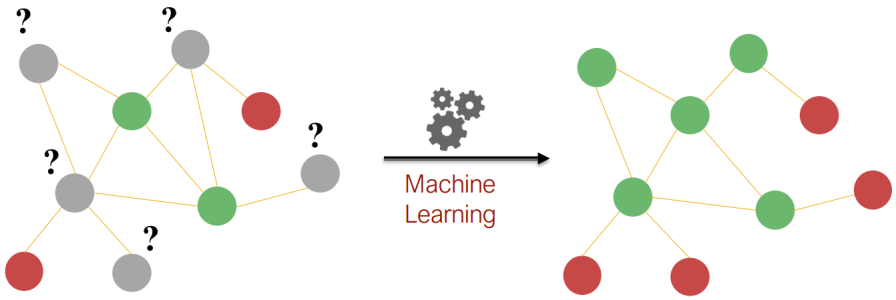


Figure 2.4: Node Classification [4]

Predicting a node's label in a graph is the focus of node-level tasks. Consider a semi-supervised task as shown in Figure 2.4 wherein few labels of the node are provided (colored), with the task here to classify or predict labels of other nodes(denoted with gray color) using a learning algorithm.

Edge level task

The task is to predict new links based on existing links or to classify the edge types. The former task is called link prediction, while the latter is known as edge classification. Consider a social network as shown in Figure 2.5 wherein users (A,B,C,D,E,F,G) are friends(black edges) with other users or they follow(blue edges) other users. The link prediction task predicts if there exists any link between users, whereas the edge classification predicts or classifies the edge existence, then what is the type of it (Friend or follower)

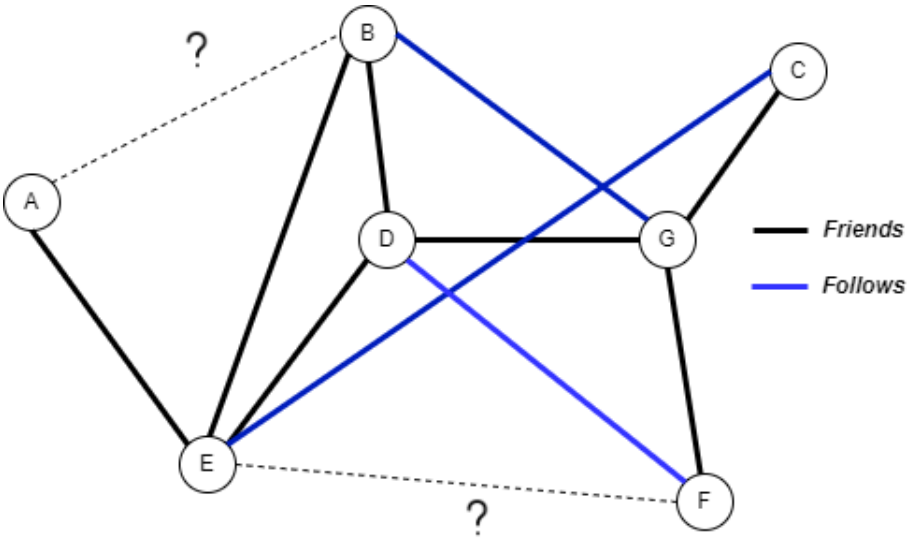


Figure 2.5: Edge level Task

[4]http://web.stanford.edu/class/cs224w/slides/02-tradition-ml.pdf

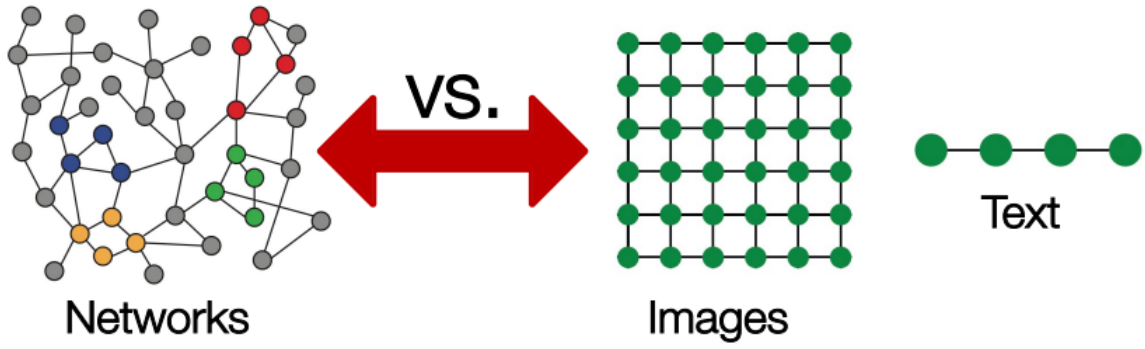


Figure 2.6: Network vs Fixed structures[5]

Graph level task

Graph-level tasks focus mainly on graph generation (e.g., generating new molecules that are similar to a given set of training molecules in a molecular graph), graph completion(e.g., predicting missing edges in a protein-protein interaction networks) or finding topologically similar subgraphs (e.g., for community detection by discovering similar cohorts). The goal here is to featurize the entire graph.

2.3.3 Challenges with graphs

Graphs are non-Euclidean data structures. This means that they do not have a fixed structure, like words in a sentence, or pixels in an image 2.6 making them complex for processing in an elegant way. Also they do not have any inherent node ordering and are extremely sparse. Recent state-of-the-art deep learning algorithms work exceptionally well on fixed structures but do not support graph structures. The representation learning on graphs discussed later in this research is focused chiefly on generalizing graph structures to a structure that advanced deep learning architectures can leverage . To summarize, the main computation challenges of graph structures are:

- Lack of consistency in structure (No fixed structure)
- Node order equivariance (No inherent ordering amongst nodes)
- Graphs are huge and extremely sparse

In general, graphs are represented in two ways:

- Adjacency matrix: For an undirected graph 2.7, an adjacency matrix Table 2.3 represents the connection between any two vertices (v_i, v_j), a presence of a connection is labelled as "1" otherwise it is "0". In a weighted graph a connection is represented by the respective edge weight. An undirected graph forms a symmetric matrix, and a directed graph an asymmetrical matrix.
  - Advantages: Easy to implement
    Edge retrieval & update takes O(1), as it can be accessed directly using the indices of the two nodes representing the edge.

[5]http://web.stanford.edu/class/cs224w/slides/02-tradition-ml.pdf

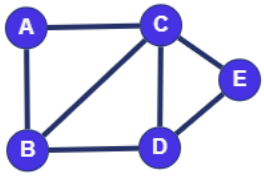


Figure 2.7: Undirected graph

	A	B	C	D	E
A	0	1	1	0	0
B	1	0	1	1	0
C	1	1	0	1	1
D	0	1	1	0	1
E	0	0	1	1	0

Table 2.3: Adjacency Matrix

Queries like whether there is an edge between v_i, v_j are efficient and can be done in O(1).

– Disadvantages: Space inefficient consumes $O(V^2)$

- Adjacency list is an array of lists wherein each entry is the list of adjacent vertices of v_i, and the array's length is the total number of vertices in the graph. The Figure 2.8 shows the adjacency list for the undirected graph Table 2.7

  – Advantages: Space complexity is O(|V|+|E|)
  Adding a new node is faster

  – Disadvantages: Queries of edge retrieval & manipulation require O(V).

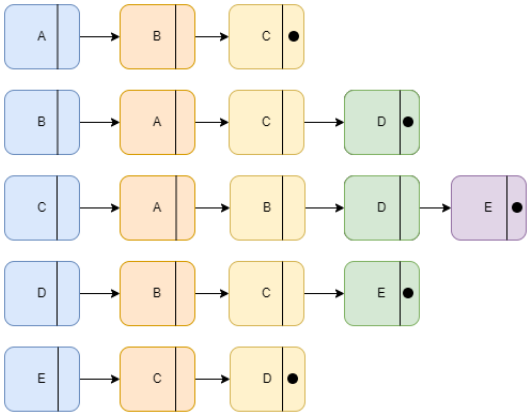


Figure 2.8: Adjacency list

For anyone with a basic understanding of neural networks, the idea of joining feature matrices with the adjacency matrix as described in figure 2.9 and feeding it to a neural network might look like a good solution, but it has two issues, as discussed in the main challenges. First the number of parameters will be really large (O(|V|)) and secondly Node order will change the predicted output (Node order equivariance)

2.4 Graph Databases

Although relational and non-relational databases have their own benefits, EHRs have traditionally been stored in relational databases such as MySQL or PostgreSQL because

[6]http://web.stanford.edu/class/cs224w/slides/06-GNN1.pdf

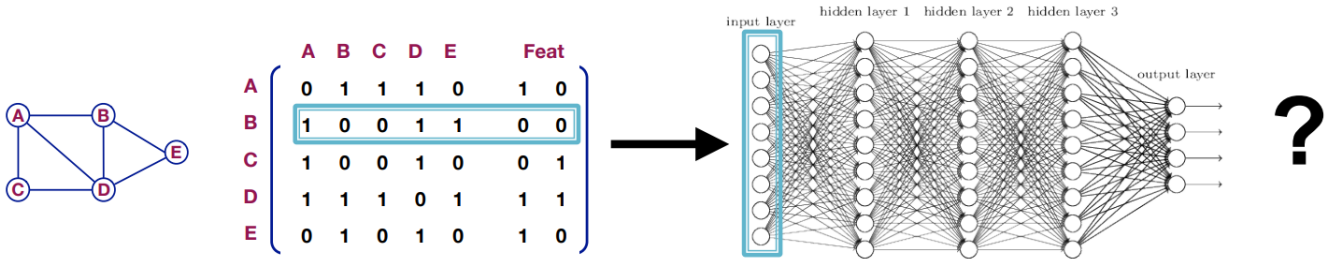


Figure 2.9: Feeding Adjacency matrix with features to MLP [6]

the data with established relationships fit easily into these databases [SN20]. In contrast, graph databases (NoSQL) are relegated to the analysis of social networks or traffic networks as they are highly connected and place a higher emphasis on relationships. The notion of relationship in relational databases is different from that of graph databases. In relational databases, the relationship is focused on the columns of the tables, whereas in the latter, it is between the data points themselves. A case study conducted by Jessica et al [SN20]. on the MIMIC-III dataset to show if *Neo4j a graph database can replace PostgreSQL in health care*, shows that although Neo4j is time intensive to implement, its cypher queries are less complex and have faster run-time for the comparable queries in PostgreSQL which can be inferred from the experiment shown in Figure 2.10. The study concludes that PostgreSQL is an adequate database, but Neo4j can be considered a viable solution for storing and analyzing healthcare data.

Show all of the patients diagnosed with Hypertension NOS					
Language	**Query**	**Number of Clauses**	**Runtime**	**DB Disk Size**	**Max DB Memory Usage**
SQL	SELECT * FROM diagnoses_icd dia INNER JOIN admissions adm ON dia.hadm_id = adm.hadm_id AND dia.icd9_code = '4019' INNER JOIN d_icd_diagnoses dicd ON dia.icd9_code = dicd.icd9_code;	7	158 ms	152 MB	4 MB[a]
Cypher (unmodified db)	MATCH (adm:Admissions)-[r:ShareHADMID]-(dia:Diagnoses_ICD {icd9_code: '4019'})-[r2:ShareICD9Code]-(dicd:D_ICD_Diagnoses) RETURN adm, r, dia, r2, dicd;	2	38 ms	682 MB	1 GB[b]
Cypher (modified db)	MATCH (adm:Admissions)-[r:DiagnosedWith]-(dicd:D_ICD_Diagnoses {icd9_code: '4019'}) RETURN adm, r, dicd;	2	21 ms	349 MB	1 GB[b]

Figure 2.10: Example comparison of command complexity and execution time in PostgreSQL (SQL) and Neo4j (Cypher)

NOS: nitric oxide synthase
[SN20]

According to DB-engines, which ranks the databases using the following criteria:

- The number of times the database has been cited on websites is determined by the number of search engine results.
- The number of searches in Google Trends indicates broad interest in the system.
- Technical discussions about system-related questions and the number of interested users on Stack Exchange are used for trends.
- The number of job postings that mention the database system is taken from Indeed and Simply Hired job platforms.
- The number of profiles in professional networks that mention the system. DB-Engines makes use of LinkedIn.
- For social network relevance The number of Twitter tweets that mention the system is counted by DB-Engines.

2.11 shows that the popularity of graph databases has drastically increased in the past decade, which implies that more and more applications are using Graph databases and making machine learning on graphs an important area of research.

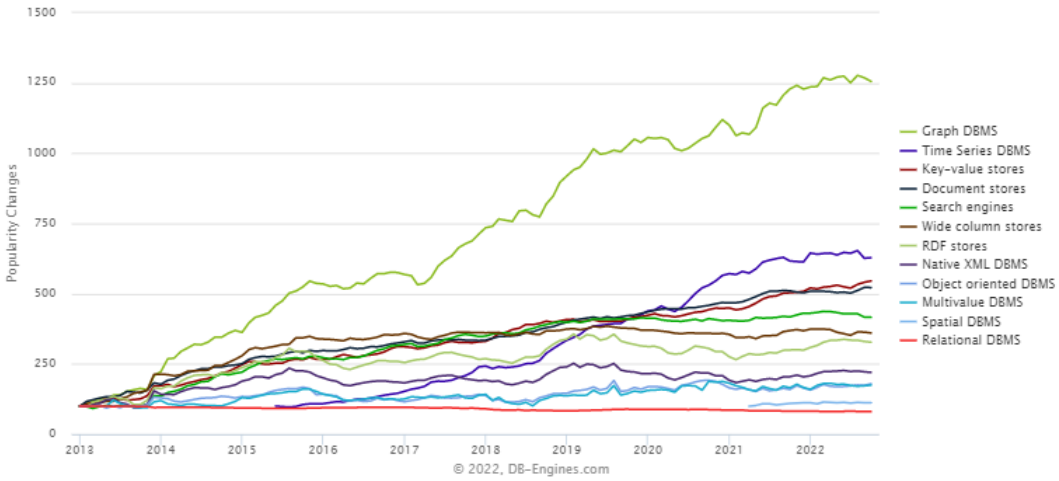


Figure 2.11: Database ranking based on popularity

2.5 Representation Learning

Any traditional machine learning task follows a certain process depicted in figure 2.12. The raw data is preprocessed, then features are extracted & selected as per the domain-specific question and passed down to a downstream machine learning algorithm. The process of feature extraction and selection is referred to as feature engineering. *Feature engineering* is an expensive & time-consuming process which requires domain expertise. Also, these features determine how good our predictions can be. Fortunately, as discussed in the 2.3.1, graphs share the same underlying schema for any type of data.The notion of automating the process of feature engineering is known as *Representation learning.*

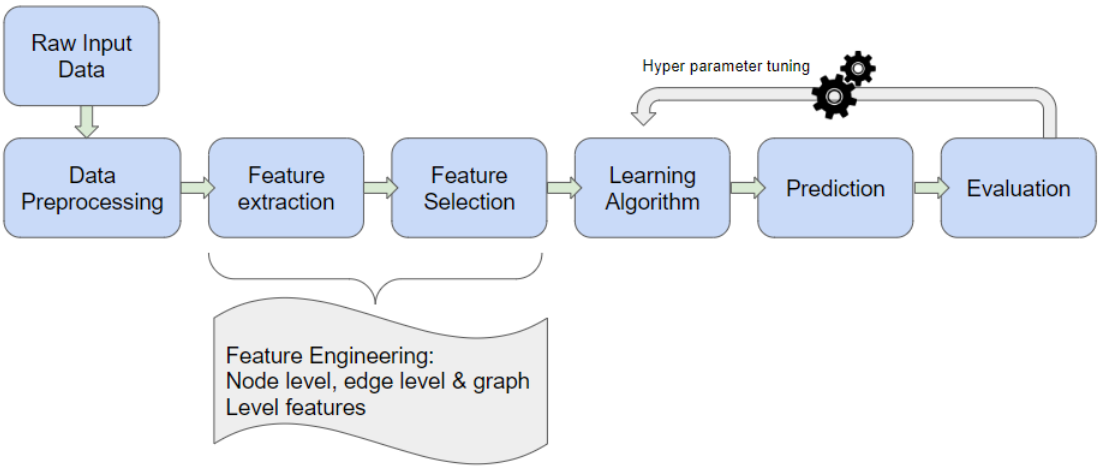


Figure 2.12: ML Pipeline

The structure and position of a node in the network can be characterized using specific measures like Node degree, Node centrality, clustering coefficients and Graphlet degree vector [Ham20]. These are sub-classified into importance-based 2.5.1 and structure-based algorithms 2.5.2. These are considered traditional approaches for analyzing graphs as they are easy to use and understand and are based on statistics. However, the downside is that they are time-consuming & expensive.

2.5.1 Importance-based

Importance-based algorithms captures the importance of the node in the network. These are useful for predicting influential nodes in the network; for example, predicting celebrity users in social networks.

Node degree

Degree k_v Counts the number of neighbouring nodes in an undirected graphs. In directed graphs, we count the edges coming in and going out of the node of interest and term them as in-degree and out-degree respectively. For example in the ,Figure 2.13 the node degree for the node k_D is 4 as there are four undirected edges through it. The drawback of Node degree as a feature is that it treats all the neighbouring nodes equally, but in reality, different neighbouring nodes might have different importance with which they contribute to the node of interest.

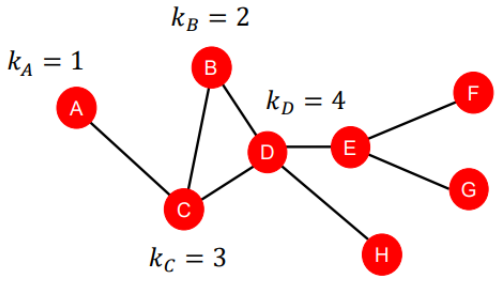


Figure 2.13: Node Degree [7]

Node Centrality

Since Node degree does not capture node importance in a network, we move to centrality measures which can capture this importance. There is a multitude of ways to model the importance of a node:

- Eigenvector centrality: A node v is important if it is surrounded by important neighbouring nodes.

$$C_v = \frac{1}{\lambda} \sum_{u \epsilon N(v)} C_u,$$

  where λ is the normalization constant (e.g., it will be the largest Eigenvalue of A), C_v eigenvector centrality of node v, N is the neighbourhood.

- Betweenness centrality: The node's importance is decided if it lies on many shortest paths between other nodes.

$$c_v = \sum_{s \neq v \neq t} \frac{shortest\ path\ between\ s\ and\ t\ that\ contains\ v}{shortest\ path\ between\ s\ and\ t}$$

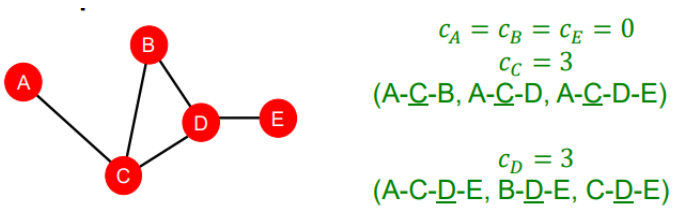


Figure 2.14: Example for Betweenness Centrality [7]

- Closeness centrality: It calculates the shortest path lengths to every other node in the network.

$$c_v = \frac{1}{\sum_{u \neq v} Shortest\ path\ length\ between\ u\ \&\ v}$$

[7] http://web.stanford.edu/class/cs224w/slides/02-tradition-ml.pdf

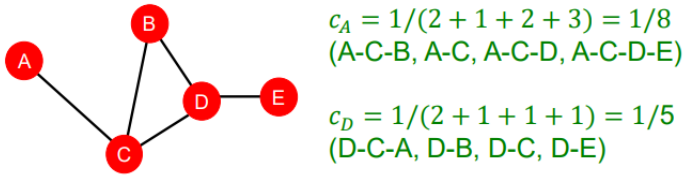


Figure 2.15: Example Closeness Centrality [8]

- PageRank: it uses the number of incoming connections and the importance of the source nodes to figure out how important each node is in a graph. The underlying assumption is that a node (like a web page) is only as important as the nodes that link to it. Mathematically, PageRank is calculated as follows:

$$PR(A) = (1-d) + d(\frac{PR(T_1)}{C(T_1)} + ... + \frac{PR(T_n)}{C(T_n)})$$

[9] where A is a node (e.g., a web page) and pages T_1 to T_n point to it, $C(T_1)$ is defined as the number of outgoing links from that page, d is a damping factor set between (0-1) default is 0.85. The algorithm has a few special cases which need to be considered.

  - If there are no relationships between pages inside a group, the group is classified as a spider trap wherein a group of pages point only to each other and assign high rank to those pages
  - When a network of pages forms an endless loop, the rank sink can arise.
  - Dead-ends arise when a page has no incoming relationships.

  Changing the damping factor can assist with all of the aforementioned problems. It can be viewed as the likelihood of a web surfer sometimes jumping to a random page and not getting caught in sinks.

  There are other centrality measures such as Bonacich centrality, distance weighted reach and many more [10].

2.5.2 Structure Based Importance

It captures the topological properties of the local neighbourhood around a node and are useful in predicting the role played by a particular node in the graph(e.g., predicting protein functionality in a protein-protein interaction graph).

- Clustering Coefficient: It is a measure of how connected node *v's* neighbouring nodes are. It is defined as :

$$e_v = \frac{number\ of\ edges\ among\ neighbouring\ nodes}{k_v/2} \epsilon [0,1]$$

[8] http://web.stanford.edu/class/cs224w/slides/02-tradition-ml.pdf
[9] https://neo4j.com/docs/graph-data-science/current/algorithms/page-rank/
[10] https://bookdown.org/markhoff/social_network_analysis/centrality.html#distance-weighted-reach

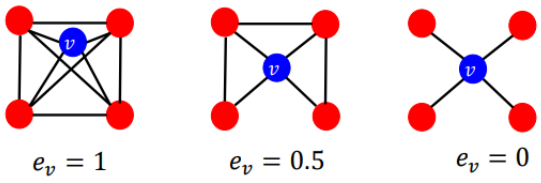


Figure 2.16: Example Clustering coefficient [11]

Clustering coefficients count the number of triangles in a graph. This is important in social networks because it helps recommend friends (e.g., new friendships between friends of someone).

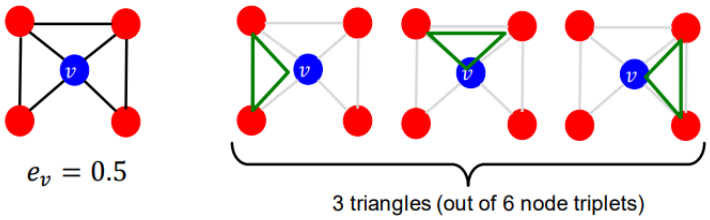


Figure 2.17: Example of Counting triangles [9]

- Graphlet Degree Vector (GDV): It is a count vector of graphlets rooted at a given node. Graphlets are small non-isomorphic subgraphs that define the structure around a node in the network. The concept of graphlets is a generalization of counting triangles, as discussed in the above method. Two graphs *(G, H)* are said to be isomorphic $G \cong E$ if there exists a bijection

$$\phi : V(G) -> V(H)$$

such that

$$u, v \epsilon E(G)$$

if and only if

$$\phi(u)\phi(v)\epsilon E(H)$$

where V is Vertex, E is an edge, and ϕ is a function that maps the Graph G & H. 2.18 depicts the graphlets. A graph with just two nodes can form a single graph, and the node can occupy either one of the positions as the graph is symmetric, whereas for the graph with three nodes, there exists two different representations (G1 & G2), and the node in G1 can occupy the positions 1,2, but in G2 it can occupy either of the three positions. Similarly, for a graph with five nodes, there exist 73 different graphlets.

[11] http://web.stanford.edu/class/cs224w/slides/02-tradition-ml.pdf

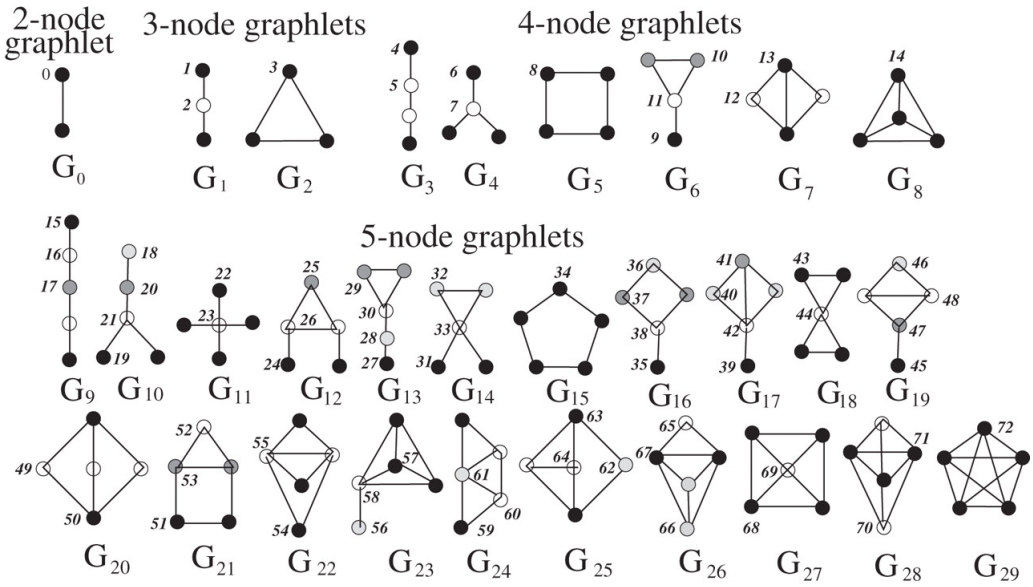


Figure 2.18: Graphlets for 5 node [Pržo7]

2.6 Random Walk

The strategy of random walks can be compared to strolling. Select a node at random and, with some probability, move to the next node and repeat this for a fixed number of steps.

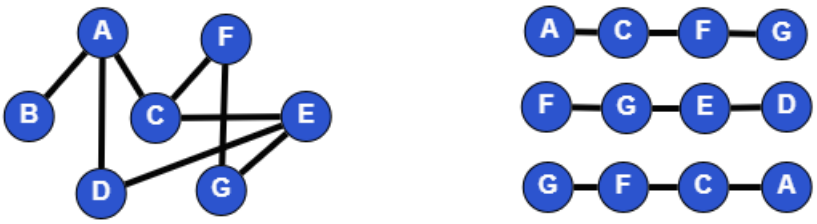


Figure 2.19: Knowledge Graph and Random walk sequence

The random walk sequence generated can be understood in fig 2.19. In the first sequence, the walk starts at node A and randomly walks through nodes C, F, and G, whereas in the second sequence, it starts at F and walks through G, E, and D. The walk length is 4.

2.6.1 Deep Walk

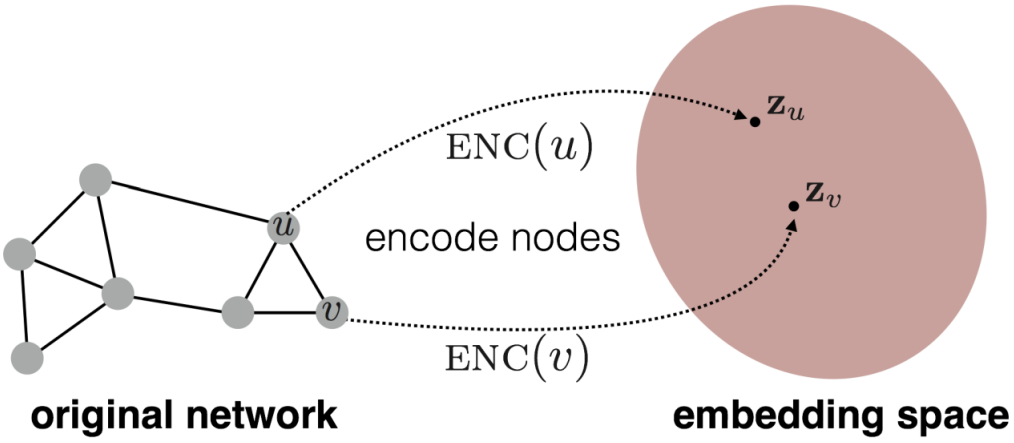


Figure 2.20: Encoding nodes to embedding space [12]

Transferring the graph structure into a numerical representation as shown in Figure 2.20 that can be fed into conventional machine learning methods is a significant challenge when working with graphs. The random walk strategy discussed forms the basis for the deep walk [PARS14] and node2Vec [GL16], In DeepWalk, the word2vec strategy is used to learn the node representations in a graph by treating each node as a "word" and each random walk as a "sentence". The word2vec algorithm is then applied to the sequences of nodes generated by the random walks in order to learn embeddings for each node.. Here is a quick explanation of the word2Vec model.

2.6.2 Node2Vec

This node embedding generation model is very much influenced by the Deep walk method. The main difference here is the strategy used for generating the random walks. In the case of the deep walk, it is random, but in the case of Node2Vec, It introduces a bias in sampling the next node in the walk denoted by α which is controlled by two parameters, p and q, that decide the likelihood of immediately revisiting a node in the walk (p) and the likelihood of moving away from a node (q). These parameters allow for the exploration of the local and global neighborhood of a node, respectively. Although the easiest way to sample the next node would be to go through static edge weights, this does not account for the graph structure. Thus, BFS & DFS are used as they account for graph structure and homophily.

In Node2Vec, the probabilities of moving from one node to another are calculated as follows:

$$P(c_i = x | c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z}, & \text{if } (v,x) \epsilon E \\ 0, & \text{otherwise} \end{cases}$$

[12] http://web.stanford.edu/class/cs224w/slides/03-nodeemb.pdf

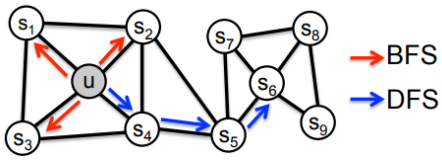


Figure 2.21: BFS and DFS search strategies from node u (k = 3) [GL16]

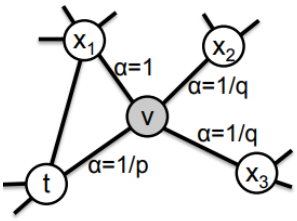


Figure 2.22: Illustration of the random walk procedure in node2vec. The walk just transitioned from t to v and is now evaluating it is next step out of node v. Edge labels indicate search biases [GL16]

where u is source node, c_i denotes ith node in the walk, l is random walk fixed length, vx is the unnormalized transition probability between nodes v and x, Z is the normalizing constant and E denotes all edges.

The transitioning probabilty π_{vx} on edges(v,x) from v is calculated as

$$\pi_{vx} = \alpha_{pq}(t, x) \cdot w_{vx}$$

where

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p}, & \text{if} d_{tx} = 0 \\ 1, & \text{if} d_{tx} = 1 \\ \frac{1}{q}, & \text{if} d_{tx} = 2 \end{cases}$$

In the case of DeepWalk & Node2Vec, structural information provided in graphs may be encoded to describe the relations between entities and provide more potential insights beneath the data [ZTXM19]. However, graphs, along with the structural information, can contain attributes such as textual data and image data on the nodes and edges (a complicated structure) [BLM+06], thus makes it more challenging to get a fundamental understanding of the information underlying the graphs. Although the structural complexity of this issue is tackled by the above-discussed embedding approaches, which include the learning of graph representations in a low-dimensional Euclidean space [ZTXM19]. When the low-dimensional representations are learnt, it is possible to quickly solve various graph-related tasks, such as the traditional node classification and link prediction [GL16]. Despite these advantages, embedding approaches suffer from a number of drawbacks due to the shallow learning mechanisms. For example, these methods might not able to capture more complex and nuanced relationships between nodes in the graph. For example, if the graph has a hierarchical structure, or if there are multiple types of relationships between nodes, shallow encoding may not be able to fully capture these relationships, resulting in suboptimal node embeddings [PARS14].

In many areas of computer vision and natural language processing, Deep Learning architectures outperform traditional machine learning methods [GDDM14]. The following section discusses the usage of deep learning architectures in the context of graphs.

2.7 Graph Neural Networks (GNN's)

Many of the graph neural network architectures are highly influenced by the existing state-of-the-art architectures in the areas of Natural Language Processing (NLP) (attention mechanism) & Image recognition such as Convolutional Neural Network (CNN). The idea is to use the power of these well-established models in the context of graphs. All the models discussed in this section follow a message passing step and aggregation step on a computation graph. A computation graph of a node is defined by its local neighbourhood.

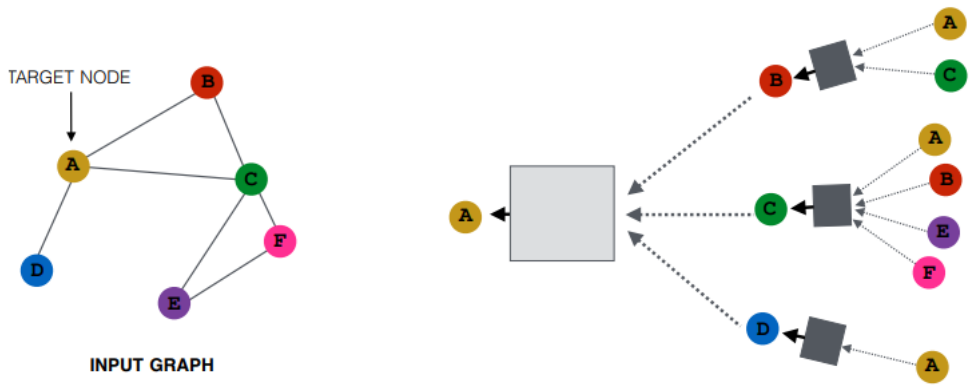


Figure 2.23: Computation Graph [13]

The design space or general framework of GNN's consists of five components as shown in Figure 2.25

- *Message*: It is a vector of features which are present on nodes and edges. In the context of EHR's, it can be demographic data (e.g.,age, gender, ethnicity) on the patient node or laboratory values on the edge connecting Admissions and Labs. Message computation can be mathematically defined as follows:

$$\mathbf{m}_{\mathbf{u}}^{(\mathbf{l})} = MSG^{(\mathrm{l})}\left(h_{\mathrm{u}}^{(\mathrm{l\text{-}1})}\right)$$

  where $\mathrm{h}_{\mathrm{u}}^{(\mathrm{l\text{-}1})}$ is the representation of the node at layer $(l-1)$, MSG is the message transformation function and $m_{\mathrm{u}}^{(\mathrm{l})}$ is the transformed message of node at layer l.

- *Aggregation*: It is the process of aggregating all the features (Messages) passed by neighbouring nodes with the target node. It can be mathematically formulated as follows:

$$h_{\mathrm{v}}^{(\mathrm{l})} = AGG_{(l)}\left(\left\{m_{\mathrm{u}}^{(\mathrm{l})}, u \epsilon N(v)\right\}\right)$$

  where AGG can be an order invariant aggregation such as sum, mean, or max. $h_{\mathrm{v}}^{(\mathrm{l})}$ is the aggregated node representation of of node v. The message passing & aggregation

[13] http://web.stanford.edu/class/cs224w/slides/07-GNN2.pdf

together form a single GNN layer. The Figure 2.24 shows all the possible components a typical GNN layer (GAT) consists of

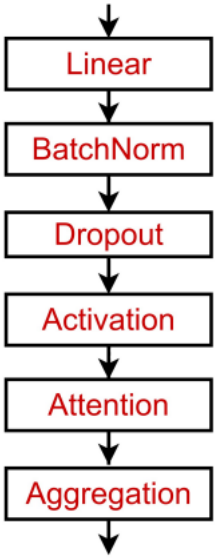


Figure 2.24: Suggested GNN layer [14]

- *Layer Connectivity*: it refers to stacking multiple GNN layers together to get the information from further away. it can be stacked sequentially or with skip connection or with any other strategy.

- *Graph augmentation*: It refers to the process of modifying the graph structure, or node /edge features, in order to improve the performance of the model. For e.g., extracting a relevant sub-graph from the original graph for the task at hand or generating new data samples by randomly traversing the graph (random walk) etc.

- *Learning objective*: An objective function is defined to train a GNN. This objective depends on the graph task (Node level, edge level or graph level). The GNN model learns the weight by minimizing the loss function.

[14] http://web.stanford.edu/class/cs224w/slides/07-GNN2.pdf

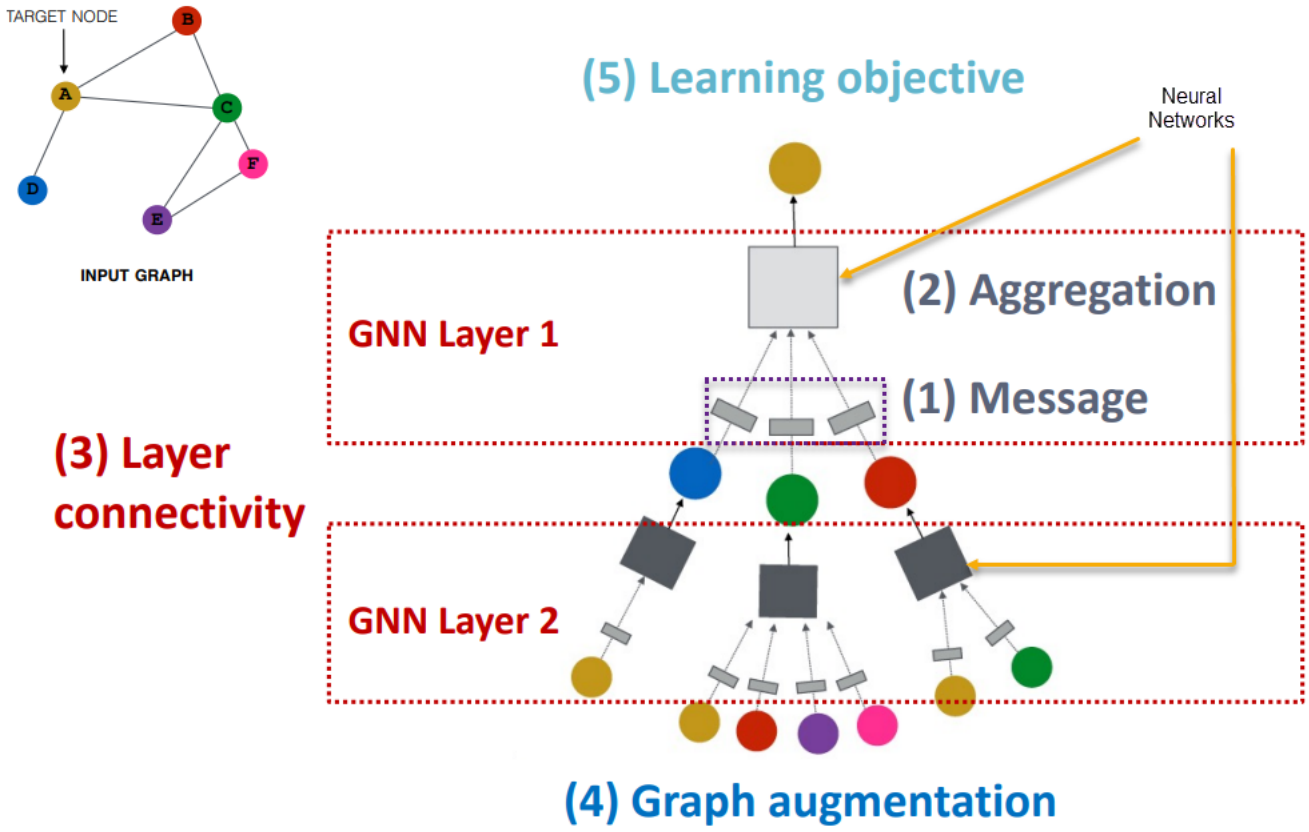


Figure 2.25: General Framework of GNN [15]

In this way, every node can have its own computation graph.

2.7.1 Graph Convolutional Network

Graph convolutional networks are one variant of GNN's, which are conceptually based on the convolutional neural networks which are used widely in the field of image recognition. In CNN's, the input image is processed in a sliding window fashion using a collection of filters (also known as kernels or weights). Each filter recognizes a certain aspect of the image, such as borders, corners, or textures. A series of feature maps, which are produced by the convolutional layer, are then processed by further layers to extract higher-level information. [GDDM14]. Further, pooling layers are added, which are designed to make the network more resilient to minute changes in the input image by reducing the dimensionality of the feature maps. The network is made more computationally efficient by pooling layers. [ZTXM19]. However, graphs have a non-Euclidean structure, thus convolutions and filtering operations performed on graphs do not provide results as clear as those performed on images [ZTXM19].

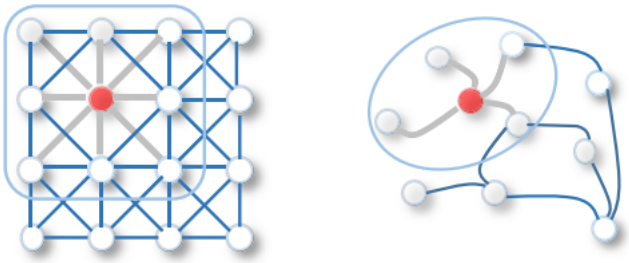


Figure 2.26: Image Convolution vs Graph convolution [WPC+20]

In the spatial domain (node domain), a graph convolution is a collection of neighbourhood node representations 2.26. A Graph Convolutional Network (GCN) follow a neural network architecture that has proven exceptional performance in a large variety of tasks and applications [ZTXM19]. These networks have a high expressive ability to understand graph representations and are able to learn them quickly [ZTXM19]. GCNs may use convolutional analysis to make use of the graph structure and collect information about nodes from the surrounding neighbourhoods [ZTXM19].
The main goal of this network is to figure out how features on a graph G= (V, E) work [KW16]. It takes the following inputs [KW16]: (A). a feature description "xi" for every node "i", that is collected into a feature matrix 'X' of the form 'N(number of nodes) X D(number of input features)'. (B). a matrix-based representation of the network structure, typically in the form of an adjacency matrix 'A'. This results in a node-level output of a 'N X F' feature matrix 'Z' (where 'F' is how many output features there are for each node) [KW16]. Then, each layer of the neural network may be represented by a non-linear function as below [KW16].

$$H^{(l+1)} = f(H^{(l)}, A)$$

, where $H^{(0)} = X$ and $H^{(L)} = Z$, L is number of layers. To better understand GCNs, consider a simple layer-wise propagation rule as an example.

$$f(H^{(l)}, A) = \sigma(AH^{(l)}W^{(l)})$$

, $W^{(l)}$ W is a weight matrix for the neural network layer (l), and $\sigma(X)$ is a non-linear activation function (e.g., sigmoid or softmax function). However, there are a few shortcomings to this straightforward model [KW16].

- When multiplying with A, all feature vectors of all neighbouring nodes are added together for each node, with the exception of the node itself. By adopting self-looping graphs (adding an identity matrix to A), this can be neutralized. It is needed because the node itself might have its own features.

- The fact that A is often not normalized means that multiplying by A will radically alter the scale of the feature vectors. To over come this disadvantage, A is normalized in the following form $D^{(-1)}A$ (Average of neighboring node features), D is the diagonal node degree matrix and $D^{(-1/2)}AD^{(-1/2)}$ (Symmetric Normalization).

By combining these two normalizations, the new propagation rule is formed as below [KW16].

$$F(H^{(l)}, A) = \sigma(\hat{D}^{(-1/2)}\hat{A}\hat{D}^{(-1/2)})H^{(l)}W^{(l)}$$

, $\hat{A} = A + I$, where I is the identity matrix and $\hat{D}$ is the diagonal node degree matrix of $\hat{A}$.

On a network graph dataset from Zachary's Karate Club, the performance of the above-mentioned GCN model is as follows.

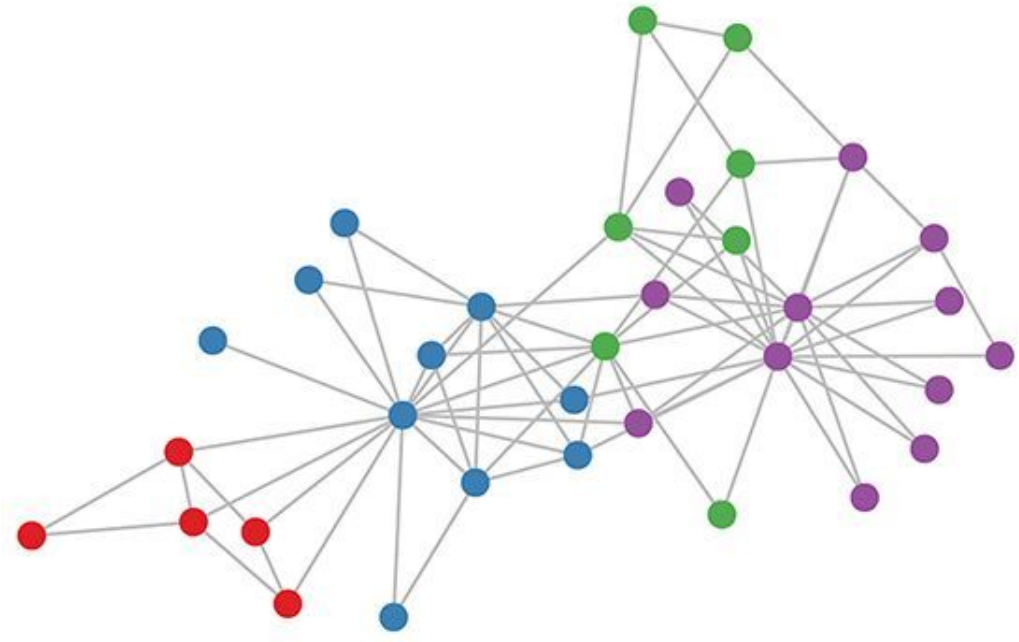


Figure 2.27: Karate club graph, colours denote communities obtained via modularity-based clustering [PARS14]

Consider a 3-layer GCN with weights that are initialized randomly. Before training the weights, we only add the identity matrix with no node features (X=I) and the graph's adjacency matrix to the model[KW16]. Now that it has three layers, the 3-layer GCN effectively convolves each node's 3rd-order neighbourhood during the forward pass. This node's embedding created by the model closely mimics the graph's community structure as sown below [KW16].

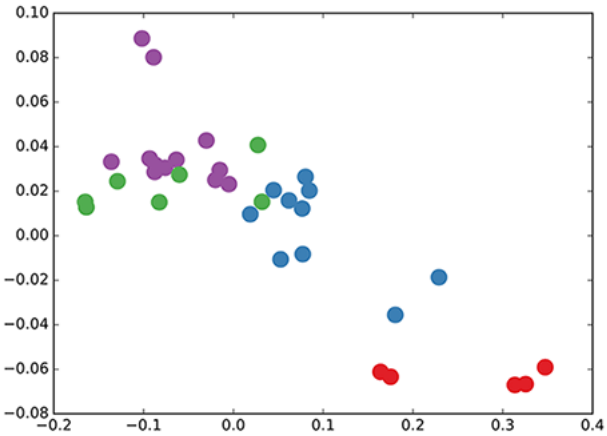


Figure 2.28: GCN embedding (with random weights) for nodes in the karate club network. [KW16]

2.7.2 Graph SAGE [HYL17a]

The embedding methods (DeepWalk, Node2Vec & GCN) are transductive in nature, which means that re-training the entire graph is necessary to generate the embedding of any new node added to the existing graph. However, re-training the entire graph is computationally expensive; thus, the idea behind GraphSAGE allows us to generate these embeddings efficiently. This implies it is efficient because it leverages node features and generalises them to unseen nodes. Unlike GCN where embeddings for each node are learnt, in GraphSAGE, a function is learnt which, if provided with the feature matrix & adjacency matrix, will return the embeddings of the node as explained in the Algorithm 2

The Application of GraphSAGE can be widely found in Social networks & Biological networks where the graphs are dynamic and large. As seen from the 2.29, GraphSAGE learns a set of *Aggregator functions* which aggregates the feature information from the node's neighbourhood for different hops such that it can be generalized to unseen nodes (inductive). The *Aggregator functions* provides the inductive capability of this model. An ideal aggregator function that ensures that the model can be trained and applied to an unordered node neighbourhood needs to be symmetric (order invariant) and, simultaneously, trainable to preserve the structural information of the graph, such as node degrees and edge weights.

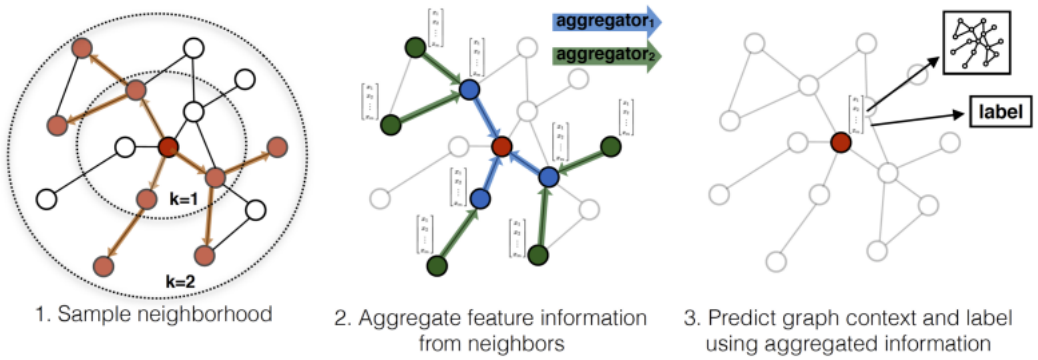


Figure 2.29: Visual illustration of the GraphSAGE sample and aggregate approach. [HYL17a]

2.7.3 Graph Attention Network

GAT is one of the popular variants of GNN's which uses an attention mechanism [VSP+17] which is widely used in NLP. In GCN, during message aggregation, all the local neighbours of the target node are treated equally. Mathematically, GCN convolution operation produces a normalized sum of neighbours as follows:

$$h_{\mathrm{i}}^{(\mathrm{l}+1)} = \sigma\left(\sum_{j \epsilon N(i)} \frac{1}{c_{ij}} W^{(\mathrm{l})} h_{\mathrm{j}}^{(\mathrm{l})}\right)$$

Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm [HYL17a]

Input: Graph G(V, E); input features x_v, $\forall v \epsilon V$; depth K; weight matrices W_k, $\forall k \epsilon 1, ..., K$; non-linearity σ; differentiable aggregator functions $AGGREGATE_k$, $\forall k \epsilon 1, ..., K$; neighborhood function $N : v \rightarrow 2^V$

Output: Vector representations z_v for all $v \epsilon V$

$h_v^0 \leftarrow x_v, \forall v \epsilon V$;
for $k = 1...K$ **do**
  for $v \epsilon V$ **do**
    $h_{N(v)}^k \leftarrow AGGREGATE_k(h_u^k - 1, \forall u \epsilon N(v))$
    $h_v^k \leftarrow \sigma(W^k \cdot CONCAT(h_v^k - 1, h_{N(v)}^k)$
  end
  $h_v^k \leftarrow h_v^k / \parallel h_v^k \parallel_2, \forall v \epsilon V$
end
$z_v \leftarrow h_v^k, \forall v \epsilon V$

where $N_{(i)}$ is the set of 1-hop neighbours, $c_{ij} = \sqrt{|N(i)|}\sqrt{|N(j)|}$ is a graph structure based normalized constant, σ is an activation function in case of GCN it is a ReLu (rectified linear unit) and W^l is the shared learnable weight matrix.

This statically normalized convolution operation is replaced by an attention weight matrix using the graph attention mechanism; this helps the model to assign importance to neighbour nodes while aggregation (figure 2.31). In order to calculate node j's importance on node i an additional attention layer is used, which can be separated into four parts as shown in figure 2.30:

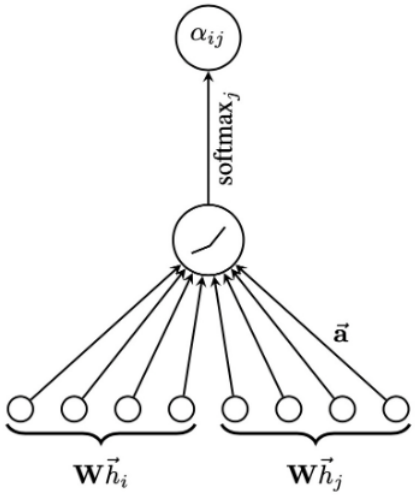


Figure 2.30: Attention Layer [VCC+17]

- Simple linear transformation: The node features of i,j are transformed to higher level features by applying a linear transformation parameterized with a weight matrix W.

$$z_{\mathrm{i}}^{(\mathrm{l})} = W^{(\mathrm{l})} h_{\mathrm{i}}^{(\mathrm{l})} \tag{1}$$

- Attention Coefficients: The transformed higher-level features (z_i, z_j) are passed as inputs to the neural network to compute the attention coefficients(unnormalized) α. The transformed features in (1) are first concatenated, then a dot product between this concatenation and the learnable weight vector is carried out, and finally, a LeakyReLu activation is carried out.

$$e_{\mathrm{ij}}^{(\mathrm{l})} = \textbf{LeakyReLu}\Big(a^{(\mathrm{l})^T}\Big(z_{\mathrm{i}}^{(\mathrm{l})} || z_{\mathrm{j}}^{(\mathrm{l})}\Big)\Big) \tag{2}$$

- Softmax: A softmax function is applied to normalize the attention scores carried out in (2) across all the nodes.

$$\alpha_{\mathrm{ij}}^{(\mathrm{l})} = \frac{exp(e_{\mathrm{ij}}^{(\mathrm{l})})}{\sum_{k \epsilon N(i)} exp(e_{\mathrm{ik}}^{(\mathrm{l})})} \tag{3}$$

- Aggregation: Finally, in the aggregation step, the embeddings are aggregated as follows:

$$h_{\mathrm{i}}^{(\mathrm{l}+1)} = \sigma\Big(\sum_{j \epsilon N(i)} \alpha_{\mathrm{ij}}^{(\mathrm{l})} z_{\mathrm{j}}^{(\mathrm{l})}\Big) \tag{4}$$

GAT employs multi-head attention as shown in figure 2.32 to increase the model's capabilities and steady the learning procedure. Here, K separate attention processes carry out the transformation defined by Equation (4), with the resulting outputs being integrated by either averaging or by concatenation as defined.

Average : $h_i^{(l+1)} = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j \epsilon N(i)} \alpha_{ij}^k W^k h^{(l)} j\right)$

Concatenation: $h^{(l+1)} i = ||k = 1^K \sigma\left(\sum j \in \mathcal{N}(i) \alpha_{ij}^k W^k h_j^{(l)}\right)$

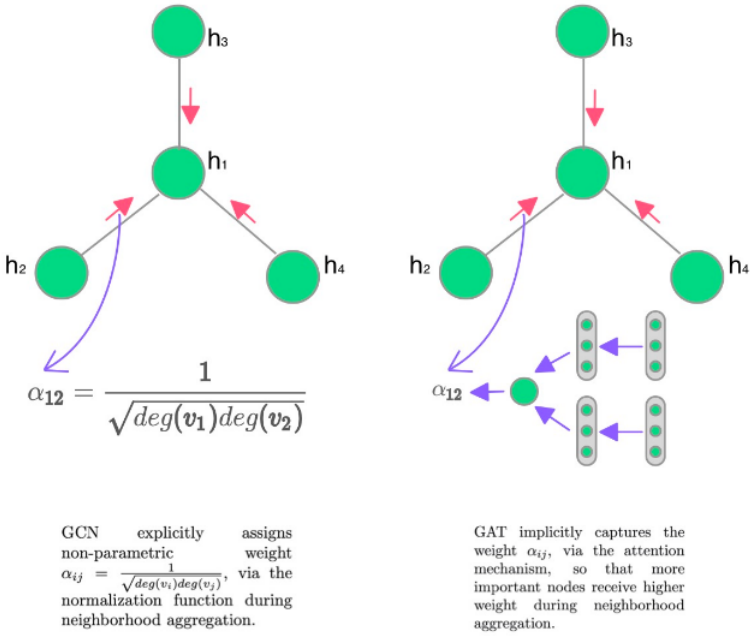


Figure 2.31: GCN vs GAT [16]

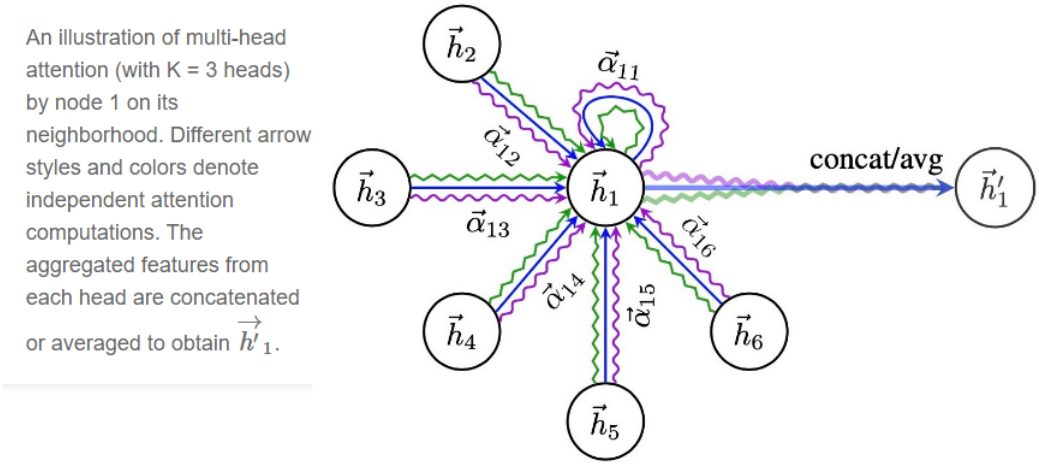


Figure 2.32: Multi Head Attention [VCC+17]

2.7.4 Challenges of Graph Neural Networks

GNN's are considered as the state of the art algorithms with many advantages over the random walk-based approaches or the traditional features extraction method, as these algorithms incorporate the features present on the nodes & edges and can process large and complex graphs. Algorithms, such as GraphSAGE, are also inductive. Nevertheless, there are also a few downsides to this algorithm. As deep neural networks are used in this algorithm, it is hard to interpret the underlying working of the model [ZTLT21]. They also run into the risk of over-smoothing, under-reaching, and

[16] https://dsgiitr.com/blogs/gat/

over-squashing [CLL+20][AY20]. Another issue with Graph neural networks is that node features are mandatory.

2.8 Text Encoding

Encoding is needed in machine learning because many machine learning algorithms require numerical input Categorical variables can contain text values which cannot be used directly. Encoding converts these categorical values into numerical values that the algorithm can use.

Encoding also helps to handle the high cardinality of categorical variables, a situation where a categorical variable has many distinct categories.

Another important reason is that encoding can improve the performance of the model by creating new features that capture the relationship between the categorical variables and the target variable. For example, target encoding will capture the relationship between the categorical and target variables.

In summary, encoding is a necessary step in preparing the data for machine learning algorithms; it allows the algorithm to understand the categorical data, helps handle high cardinality, and creates new features that improve the model's performance.

There are multiple ways in which categorical values can be encoded into numerical values. Below are some encodings which are used in this research:

- **Label encoding** is done by assigning a unique integer value to each unique category in the variable. For example, if a variable has the categories "red", "green", and "blue", label encoding would assign the values 0, 1, and 2 to these categories, respectively.

  - Advantages of Label Encoding: Simple and easy to implement.
    Takes up less memory
    Can be useful in ordinal categorical variables.

  - Disadvantages of Label Encoding: Assumes an ordinal relationship between categories, which may not be the case.
    May not be suitable for non-ordinal categorical variables.

- **One-hot encoding** is done by creating a new binary column for each unique category in the variable. Each row in the dataset will have a "1" in the column corresponding to the category it belongs to and a "0" in all other new columns. For example, if a variable has the categories "red", "green", and "blue", one-hot encoding would create three new binary columns: "is_red", "is_green", and "is_blue". The original data will be replaced with a vector [1,0,0].

  - Advantages of One-Hot Encoding: Creates binary columns for each category, so the categories are not ordinally related.
    Can be useful for non-ordinal categorical variables.

- Disadvantages of One-Hot Encoding: Increases dimensionality and memory usage, especially if there are many categories.
  Can lead to sparse data, which can be a problem for some algorithms.

- **Frequency encoding** is done by replacing the categories in the variable with the frequency or count of that category in the dataset. For example, if a variable has the categories "red", "green", and "blue", and the frequency count of these categories in the dataset are 10, 15 and 20, respectively, frequency encoding would replace the category "red" with 10, category "green" with 15 and category "blue" with 20. This method is proper when the categories are ordinal, and the categories with higher frequency are More important.

  - Advantages of Frequency Encoding:

    It can be useful when the categorical variable has a natural order based on the frequency of the values.
    It can be useful for handling categorical variables with a large number of categories.
    It can be useful for handling categorical variables with rare categories.

  - Disadvantages of Frequency Encoding:

    It can lead to information loss if the categorical variable does not have a natural order based on the frequency of the values.
    It can lead to incorrect assumptions about the relationship between the frequency and the target variable.
    It can be sensitive to the sample size and distribution, and the frequency of a category can be affected by the size of the dataset.

- **UMLS-BERT (Unified Medical Language System - Bidirectional Encoder Representations from Transformers)** [MWK+20] it is based on the BERT model, which uses a transformer-based architecture and is trained using masked language modelling. Large datasets of medical text, including electronic health records, clinical notes, and biomedical literature, were used to train this model. It is specifically designed to capture the complex and domain-specific language used in the medical field and has been fine-tuned on a large dataset of medical text. It is used for medical concept extraction, medical entity recognition, and relation extraction and has achieved state-of-the-art results on several benchmarks. UMLS-BERT is a powerful tool for natural language processing in the medical domain, and has the potential to significantly improve the efficiency and accuracy of tasks such as medical record analysis and clinical decision support.

  - Advantages of UMLS-BERT:

    It is pre-trained on a large corpus of medical text, making it well-suited for natural language processing tasks in the medical domain.
    It can improve the performance of downstream tasks such as named entity recognition, relation extraction, and question answering in the medical domain.
    It may be fine-tuned on a particular task or dataset with minimum data, hence eliminating the requirement for vast volumes of task-specific labeled data.

– Disadvantages of UMLS-BERT:

  It may not be appropriate for tasks outside the medical domain.
  It may not be suitable for tasks that require a deep understanding of medical concepts, as it is only pre-trained on a general medical corpus.
  It's a large model, so it may be computationally expensive to use and fine-tune.

3 Related Work

In this chapter, related work concerning the main research questions is discussed as follows.

- In Section 3.1, research conducted in the area of sepsis mortality prediction using traditional machine learning algorithms are discussed.
- In Section 3.2, the deep graph neural network architecture in the areas of the biomedical domain using homogeneous and heterogeneous graphs are discussed.

3.1 Traditional machine learning approaches for mortality prediction

There have been many studies in the area of mortality prediction for patients diagnosed with sepsis using traditional machine learning approaches such as random forest [TPV+16], Support Vector Machines (SVM) [RLRS+11], Extreme Gradient Boosting (XGBoost) [HLH+20]. These studies mostly focus on how machine learning models can continuously improve their performance with more data and can find non-linear relationships in the data. The studies also showed how flexible these models are and that they require less feature engineering than statistical models such as SAPS-II, SAPS-III, APACHE-II, SOFA and many more.

A study was conducted by Nianzong Hou et al. [HLH+20] on the MIMIC-III dataset for sepsis mortality prediction, which shows the comparison between three different models, namely SAPS-II, logistic regression and XGBoost. A total of 4559 patients who were diagnosed with "Sepsis", "Severe Sepsis", and "Septic shock" were included, of which 3670 survived, and 889 died. The models were evaluated on AUROC metric and showed a predictive power with AUROC of 0.819(95% Confidence Interval (CI) 0.800–0.838)) for SAPS-II, 0.797% (95% CI 0.781–0.813) for logistic regression, and 0.857% (95% CI 0.839–0.876) for XGBoost. Furthermore, they showed that urine output, lactate, BUN, sysbp, INR, age, cancer, SpO2, sodium, AG, and creatinine acted as the top 11 features in XGBoost model prediction.

Research conducted on *Early hospital mortality prediction using vital signals* by Sadeghi et al. [SBR18] using the MIMIC-III dataset showed the use of patients' Electrocardiogram (ECG) signals as input features. It was done because laboratory results of some tests require more time to be processed.

Several statistical measures such as (min, max, mean, mode) of signal-based features are passed to eight different classifiers (decision tree, linear discriminant, logistic regression, SVM, random forest, boosted trees, Gaussian SVM, and K-nearest neighbourhood (KNN). Figure 3.1 shows how well the models performed on these features.

Classifier	Precision	Recall	F1-score	Interpretability
Random forest	**0.97**	**0.97**	**0.97**	**Hard**
Gaussian SVM	0.95	0.96	0.96	Hard
Decision tree	**0.90**	**0.92**	**0.91**	**Easy**
Boosted trees	0.91	0.83	0.87	Hard
K-NN	0.80	0.85	0.82	Hard
Logistic regression	0.77	0.67	0.72	Easy
Linear discriminant	0.78	0.66	0.71	Easy
Linear SVM	0.80	0.63	0.70	Easy

Figure 3.1: Classification results for mortality [SBR18]

3.2 Graph Neural Network approaches in the bio-medical domain

Graph neural networks gained popularity in recent years with the advent of deep learning, rapidly increasing computation power and the availability of large graph datasets [WPC+20]. As explained in the background section 2.7, there are different advancements of graph neural networks such as GCN [KW16], SAmple and aggreGatE (SAGE)[HYL17a], and GAT[VCC+17]. There are many applications of Graph neural networks in the areas of bio-medicine [[ZCH+20], [JWH+21], [HYL17b]], especially in the area of mortality prediction [RTV+21], [WHA+21] with both homogeneous and heterogeneous graphs.

Some of the early works in the area of graph neural networks, such as GCN's, concentrate only on homogeneous graphs and do not consider the impact of different node types and edge types. More recent research studies have tried different approaches to handle the heterogeneity of the data.

The heterogeneous graph attention network (HAN) proposed by Wang, Xiao et al. [WJS+19] is based on GAT explained in the background section 2.7 to support the heterogeneous graphs. It is built on hierarchical attention, which includes "attention" at both node and semantic levels. The semantic-level attention is able to learn the relevance of various meta-paths. A meta-path can be understood as a specific relationship between nodes. In contrast, node-level attention focuses on learning the importance between a node and its neighbours based on meta-paths.

The Figure 3.2 shows a heterogeneous graph example for a movie dataset [3] with three different node types namely movie, actor and director and two different edge types, "ACTS" & "DIRECTS" between actor - movie and director - movie, respectively. In the IMDB dataset, movie-actor-movie or movie-director-movie acts as meta-paths.

The Figure 3.3 explains the complete architecture of the model wherein all node types are projected into unified feature space and weights of meta-path for node pairs are learnt via

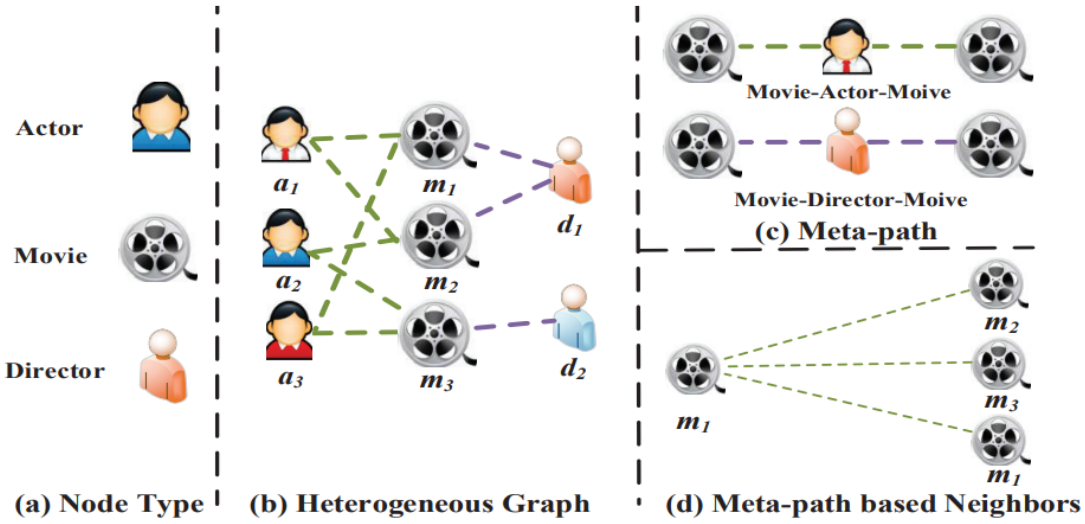


Figure 3.2: Heterogeneous graph example (IMDB) [WJS+19]

node-level attention. Then joint learns the weights of each meta-paths with the semantic-specific node embeddings via semantic-level attention, and finally, the loss is calculated, and weights are adjusted using backpropagation.

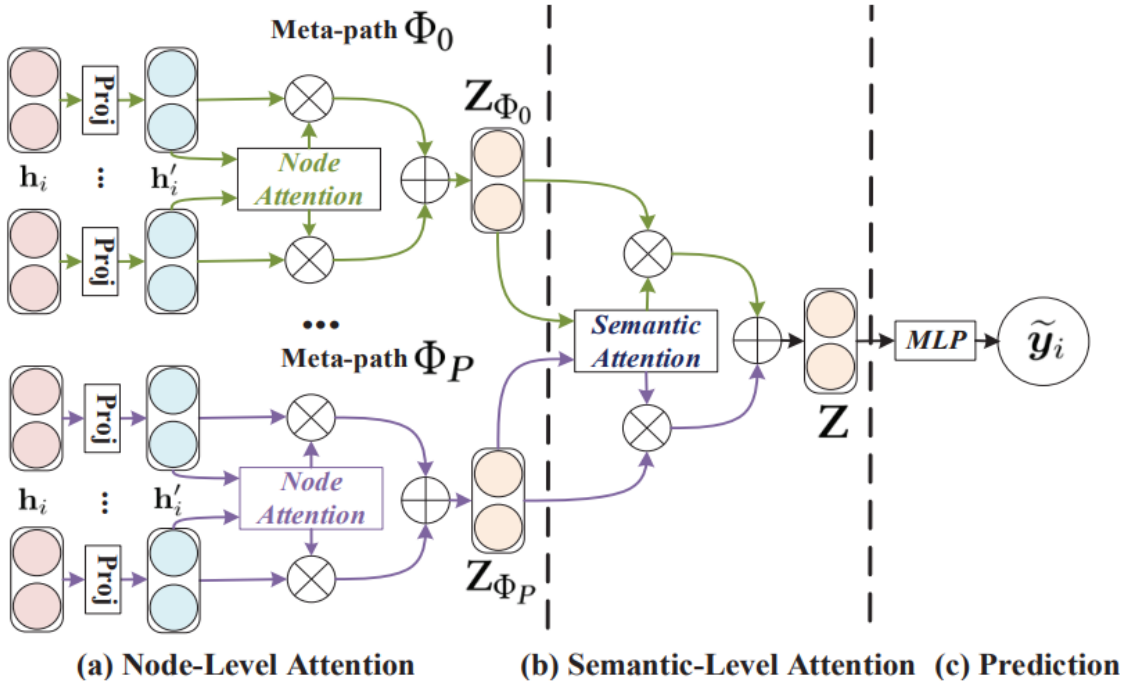


Figure 3.3: HAN architecture [WJS+19]

This model, along with two different variations HAN_{sem} (eliminates semantic-level attention and assigns similar significance to each meta-paths),HAN_{nd}(eliminates node-level attention and assigns similar significance to each neighbour) was evaluated on Macro-F1 metric with different heterogeneous graphs datasets such as Database Systems and logic

programming (DBLP) [1], Association for Computing Machinery (ACM) [2], Internet Movie Database (IMDB) [3] and found to outperform some benchmarking models such as Deepwalk (Macro F1: 84.17%), metapath2Vec (Macro F1: 73.8%), GCN (Macro F1: 88.29%), and GAT (Macro F1: 87.33%).

A research study on *Heterogeneous Similarity Graph Neural Network on Electronic Health Records* by Zheng Liu et al. [LLP+20] using MIMIC-III dataset converts the heterogeneous graphs to similarity subgraphs which are homogeneous using meta-paths. Then these subgraphs are fed as input to the graph neural network (GCN & GAT). The HAN model discussed above acts as a baseline for this model.

Figure 3.4 shows the complete architecture of the model. The raw input graph is converted to similarity sub-graphs by calculating a Symmetric PathSim(SPS), which is used to measure the node pairs with each meta-path such as Visit-Diagnosis-Visit (V-D-V) or Medication-Visit-Patient(M-V-P) as can be seen in the preprocessing step of the figure. The first and second visits of the patient have one diagnosis common between them out of the four diagnoses. As a result, the SPS between the patient's first and second visits based on the diagnosis is 2/4. Then each similarity subgraph, along with node features, is passed to different graph neural networks such as GCN, which only handles homogeneous graphs to fuse them together, preserving the true relations between the node pairs.

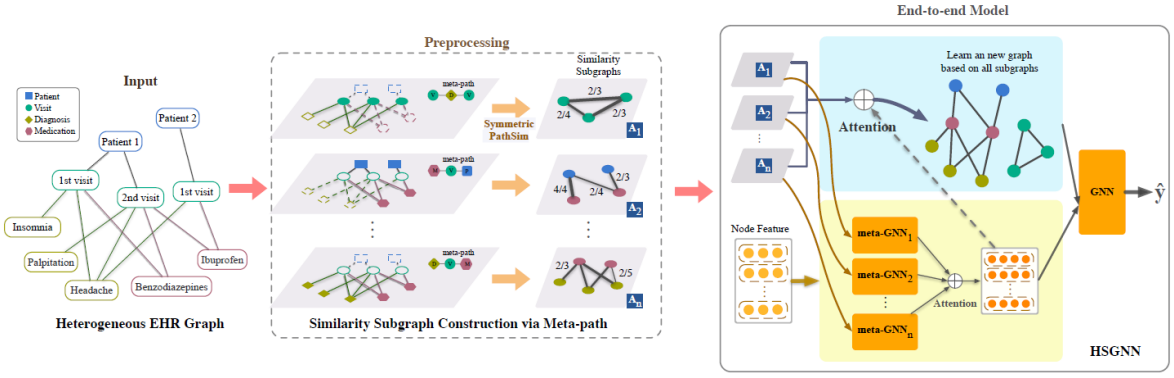


Figure 3.4: Architecture of HSGNN [LLP+20]

The model was used for diagnosis prediction on the MIMIC-III dataset and evaluated on precision metric and performed well (0.8189) against the baseline model such as HeteroMed (0.7866) [HCW+18], HAN (0.8083) [WJS+19], HetGNN (0.8070)[ZSH+19], GCT (0.8107) [CXL+20] which also support the heterogeneous graphs.

[1]https://dblp.uni-trier.de/
[2]http://dl.acm.org/
[3]https://www.imdb.com/interfaces/

4 Dataset

A dataset is a collection of information relevant to the question in the picture. It is needed to train the machine-learning algorithm to make predictions. In the context of this thesis, the collection of EHR's of actual patients collected over 10 years from a hospital in the US is considered as it has been used in many research studies and is widely considered a benchmark. Although a real-world dataset is used in this thesis; access to such datasets are hindered due to legal constraints, privacy concern, security and intellectual rights making the accessibility time-consuming and difficult [BQE+21].

Alternatively, due to these constraints, many researchers tend to use synthetic datasets. One such dataset in the health care domain is the Synthea dataset[WKN+18]. The algorithm used in Synthea aims to generate a large number of electronic health records for synthetic patients based on the ten most frequent primary and chronic conditions with the highest mortality rates in the US. Although synthetic datasets overcome regulatory restrictions, streamline simulation, enable easy manipulations, & avoid common statistical problems such as data imbalance [BQE+21], they depend on the underlying data or information used for generating them, which leads to colossal bias and also makes the results from these datasets associated with high scepticism for credibility.

4.1 MIMIC-III Dataset

MIMIC-III (Medical Information Mart for Intensive Care) is a collection of patients' health records who were admitted to the critical care units of Beth Israel Deaconess Medical Center, Boston, Massachusetts between 2001-2012. The patient information is deidentified following the Health Insurance Portability and Accountability Act (HIPAA) to respect privacy concerns. The data was collected using the CareVue & MetaVision clinical information systems. A clinical information system (CIS) is intended for usage in a critical care setting, such as an ICU. It can be a network of all the computers in a modern hospital, like those in the pathology and radiology departments. It takes information from all these systems and puts it into an electronic patient record that clinicians can look at while they are with the patient. The complete dataset can be accessed by becoming a credentialed PhysioNet user and completing (CITI) program's "Data or Specimens Only Research" training. This training is required to ensure compliance with regulations, data security and privacy, to consider ethical considerations, to follow best research practices, and to promote responsible conduct of research when handling human data and specimens. For more information, please follow this: https://physionet.org/content/mimiciii/1.4/

MIMIC-III dataset has been used for benchmarking many tasks in the healthcare domain, such as mortality prediction [LXZ+21], length of stay prediction [GAD+17], medical code (ICD) prediction [BDLP20], multivariate time series analysis [CPC+18], and biomedical

CSV file names	Description	Considered for experiment
PATIENTS	Information specific to Patient Subject id, date of birth, date of death Gender	✓
ADMISSIONS	Information specific to Admission date of admission, ethnicity, religion, reason of admission,	✓
DIAGNOSES_ICD	ICD-9 Code, Long title, Short title	✓
PRESCRIPTIONS	drug type, drug names, dosage value, dosage unit, start date, end date, strength	✓
OUTPUTEVENTS	Type of excretion, values, Unit of measurement, charttime, stopped, new bottle	✓ only Urine output
NOTEEVENTS	Clinical note text, category, chart time, description	✗
LABEVENTS	Lab name, type, value, unit of measurement,flag	✓
INPUTEVENTS	Input type name, amount, unit of measurement, route of feeding, stopped, new bottle, chart time	✗
CHARTEVENTS	Vital name, value, unit of measurement, warning error, result status, chart time	✓ ["BUN", "Hgb", "Respiratory Rate", "Arterial BP [Systolic]", "CaO2", "Dialysis Type", "Diet Type", "Eye Opening", "GCS Total", "Heart Rate", "Manual BP [Systolic]", "Motor Response", "NBP [Systolic]", "O2 Flow (lpm)", "Skin [Temperature]", "SpO2", "Temperature C", "Verbal Response", "Arterial PaO2", "FiO2", "Manual Blood Pressure Systolic Left", "Manual Blood Pressure Diastolic Right", "Manual Blood Pressure Systolic Right", Manual Blood Pressure Diastolic Left", "GCS - Eye Opening", "Arterial Blood Pressure systolic", "Non Invasive Blood Pressure systolic", "GCS - Verbal Response", "GCS - Motor Response", "O2 saturation pulseoxymetry"]
MICROBIOLOGYEVENTS	Microbiology event name, specimen description, chart time	✗
PROCEDUREEVENTS	Procedure description, start time, end time, value, location	✗

Table 4.1: MIMIC-III data description and their usage in the experiment

text classification [MKB+20]. The Table 4.1 describes the different data files and their content in the MIMIC-III dataset and all the components that are considered for the experiment in this thesis.

4.1.1 MIMIC-III structure

Figure 4.1 provides a complete overview of the dataset. During the hospitalization, the patient's information is collected and categorized into 26 different tables, including admission, demographic data, laboratory results, vital signs, procedures performed, medications given, preconditions, medical notes, and many more. Before publishing the MIMIC-III database, the data is archived as a whole and then preprocessed to maintain the patients' anonymity by abstracting the essential information that can be utilized to trace the patient's identity. The steps used in this process are as follows:

- De-Identification: 18 different data elements about the patient (as per HIPAA Act) were identified and removed to maintain the anonymity of the patient, such as name, address, telephone number etc.,
- Date Shifting: The date and time for every single action performed at the hospital on the patients, such as testing for labs, prescribing drugs, performing procedures, etc., were all shifted with a random offset consistent with preserving the actual interval.
- Format Conversion: The free text present in the reports provided by physicians, radiologists, or nurses can contain sensitive information about the patient; this information was masked using exhaustive dictionary lookups and pattern matching using regular expressions.

Finally, the curated data is stored in the database and made available to the credentialed user. It is then carefully monitored and updated regularly based on database user feedback.

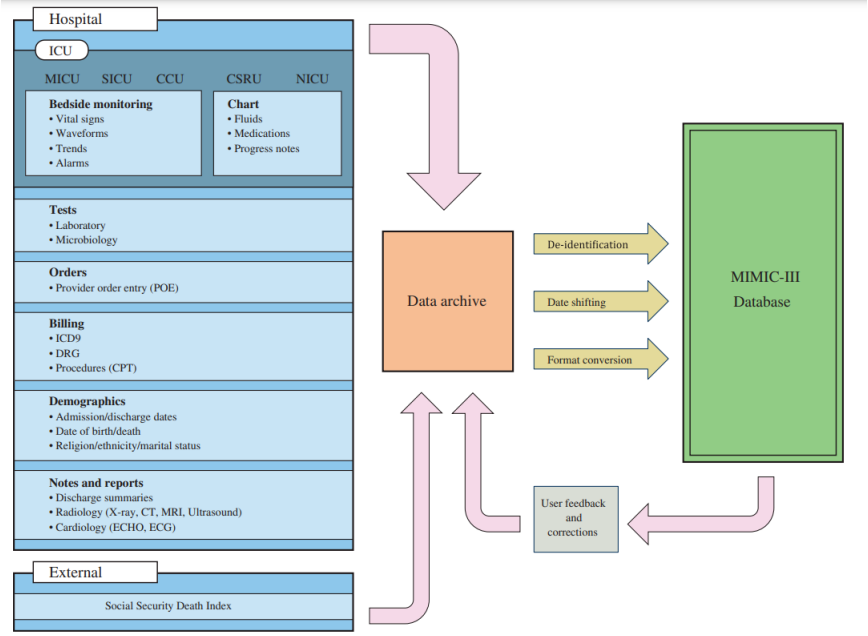


Figure 4.1: MIMIC-III database overview [JPS+16]

5 Design

The Chapter 2 of this thesis provides a comprehensive introduction to the concepts and terminology fundamental to the work. In this chapter, precise research questions are defined and addressed. An experimental design practised for mortality prediction is outlined and a detailed data analysis and data preprocessing is provided. For the sake of organization, this chapter will proceed as follows:

- Firstly in section 5.1, specific research questions are established that are evaluated in this thesis.
- In section 5.2, the design pipeline followed in this thesis is presented.
- The section 5.3 presents a detailed data analysis of the patients diagnosed with Sepsis condition in the MIMIC-III dataset.
- Finally, in section 5.4, a complete data pre-processing carried out on the dataset before passing them onto the graph neural network model is explained.

5.1 Research Questions

This thesis aims to understand how well graph machine learning models capture the complex underlying information of EHR's presented in a graph data structure. Since the data can be modelled in various formats in the graph structure itself, we try to understand how different graph data representations impact the quality of the embeddings generated from graph machine learning models. In order to evaluate the embeddings, we formulate our learning task as mortality prediction of patients diagnosed with Sepsis. Finally, we compare the important mortality predictors extracted from graph neural networks with the SAPS-II & SAPS-III model.

- **RQ1:** What will be the effect of different data representations in the graph (graph modelling) on the model's performance, GPU usage, and processing time?
  - Does different encodings of free text (i.e. Diagnosis, or Drug name) affect the predictive power of the model?
- **RQ2:** What are the important predictors of mortality for the patients diagnosed with Sepsis according to the GAT model, and how do they compare with the well-established SAPS-II & SAPS-III model predictors?
- **RQ3:** How different individual relationships contribute to the mortality prediction using GAT model?
  - how does GAT handle a biased relationship in the structure?

5.2 Design Pipeline

The Figure 5.1 explains the high-level design approach followed in this thesis. Initially, the dataset stored in CSV files is parsed, and a detailed analysis was performed. Then, this data is uploaded to a graph database (Neo4j). Now, the database is queried as per the requirement. A pre-processing step is carried out as discussed in 5.4, thus eliminating the unnecessary noise and embeds the categorical & textual data into the numerical format as machine learning algorithms are designed to work with numbers and vectors. A heterograph is created that bundles nodes, node features, edges, and edge features between the nodes together. The heterograph is passed as an data object to the graph neural network to generate node embeddings. The embeddings from the GNN's can then be passed on to the downstream machine learning algorithm for making predictions. In this thesis, the graph neural network is trained in an end-to-end fashion, i.e., the GNN can directly predict the outcome/class. The results are evaluated using a metric (AUROC), and based on the evaluation, the hyperparameters, as discussed in 6.2.2 are tuned to optimize the task.

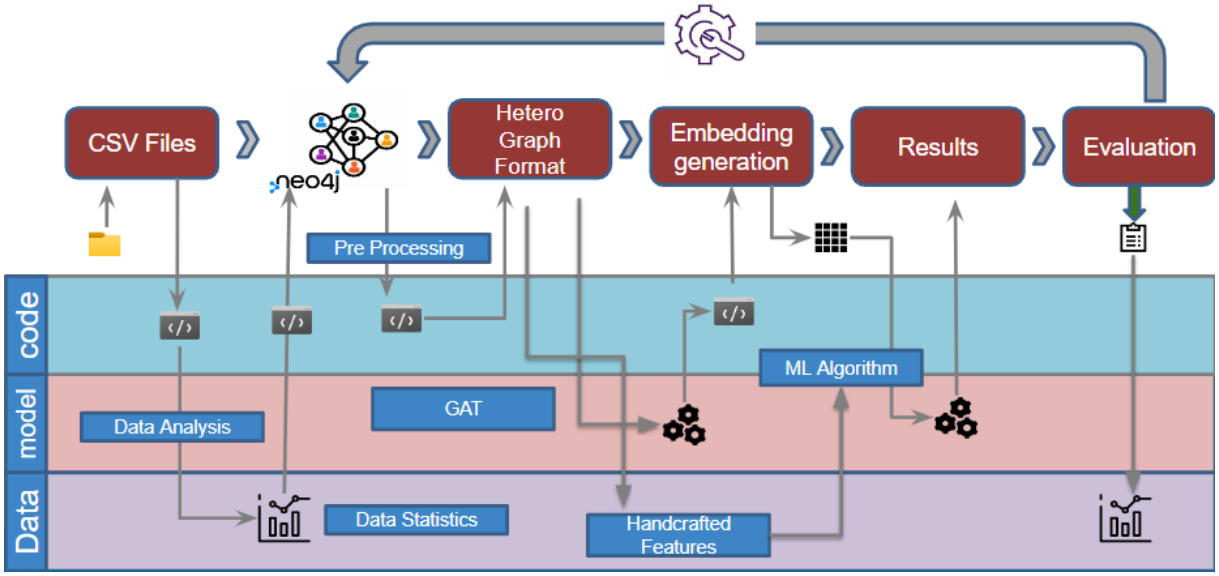


Figure 5.1: Framework high-level view

5.3 Data Analysis

This section comprises a detailed data analysis of the Medical Information Mart for Intensive Care (MIMIC) dataset concerning the patients diagnosed with Sepsis condition. In general, the Table 5.1 represents the overall and Sepsis-related statistics of the dataset. On average, the patients diagnosed with Sepsis had 39.3% more visits than those unrelated to Sepsis. The Figure 5.2, Figure 5.3, Figure 5.4, Figure 5.5 & Figure 5.6 shows the patient mortality distribution based on their biological gender and different demographic indicators such as Ethnicity, Religion, Age group & Marital status. It helps us correlate our results to Representative data. The mortality % is indicated inside each bar of the charts. The Figure 5.2 shows the mortality of the patients who had Sepsis based on the

biological genders; it can be seen from the chart that the number of patients who survived is much more than that of the expired; this shows that there is a considerable imbalance in the survived (indicated with blue colour) and dead patients(indicated with red colour), but there is no considerable imbalance between genders itself thus referring to a healthy distribution amongst both genders.

Nodes	Total number	Sepsis - Related
Patients	**46520**	**9928**
Admissions	**58976**	**15652**
Labs	**753**	**693**
Drugs	**4525**	**2893**
Diagnosis	**14567**	**4549**
Procedures	**3882**	**1225**

Table 5.1: Patients overall & Sepsis distribution

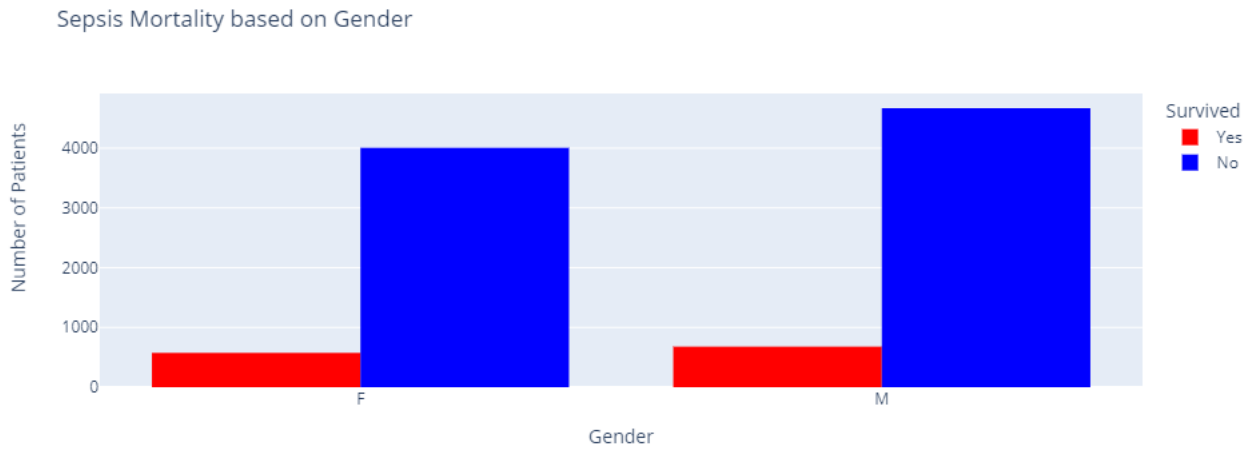


Figure 5.2: Sepsis Mortality by Gender

The figure 5.3 shows that cases of Sepsis are found in all age groups but are dramatically high in infants & kids aged ten years or below. However, the mortality in them is relatively low. The number of patients diagnosed, as well as the mortality rate, rises with age. Higher mortality is found in older adults. Since consideration of the age group 0-10 skews the data towards this group and might not be an actual representation of the general population, it is ignored in this study.

The figures 5.4, 5.5, 5.6 show the demographic distribution for ethnicity, marital status and religious identification amongst the patients. These factors also play an essential role in predicting mortality [Con13]. According to the distribution, the total number of patients diagnosed with Sepsis is predominantly white, married or single, or identified as Catholics although this might result from the huge imbalance between the different groups(e.g., X% white patients and Y% Asian patients). Thus it acts as a reference for the interpretation of the results.

The figure 5.7 shows the reason for the admission of the patient to the hospital. It plays a vital role in mortality prediction in models such as SAPS-II & SAPS-III. It also

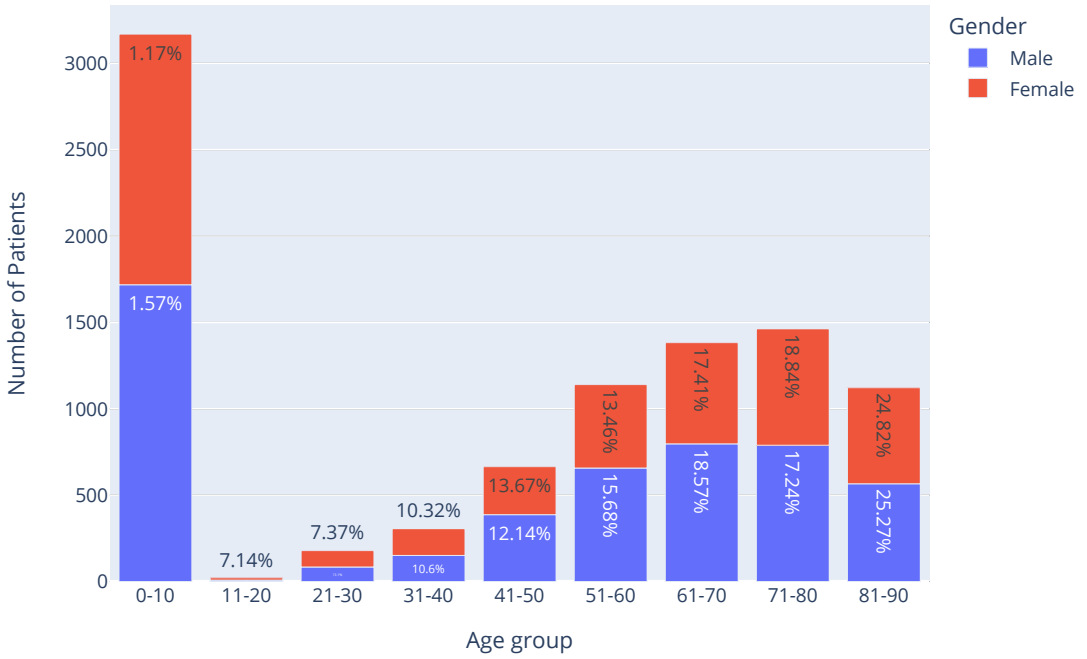


Figure 5.3: Sepsis Mortality by Age group. The percentage inside of each bar represents the mortality % for respective gender & age-group

demonstrates that emergency and newborn cases predominated among those diagnosed with Sepsis.

Figure 5.8 show different types of laboratory test categories in the dataset. There are 753 unique lab tests categorized into three main categories (Hematology, chemistry & blood gas) [Rif17].

Hematology is the study of blood and blood-forming tissues. Hematology tests are often used to diagnose and monitor various health problems, such as anemia, bleeding disorders, and blood cancers. Some standard hematology tests include:

- Complete blood count (CBC) is a blood test that examines the total number of blood cells, including red blood cells, white blood cells, and platelets.
- Hemoglobin and hematocrit: These tests measure the amount of hemoglobin and the volume of red blood cells in the blood. it is a protein in red blood cells that carries oxygen.
- Coagulation tests: These tests measure the ability of the blood to clot and are used to diagnose bleeding disorders.
- Differential white blood cell count: This test measures the number and types of white blood cells in the blood and is used to help diagnose infections and immune system disorders.

Chemistry tests are laboratory tests that measure the concentration of various chemicals in the blood. These tests diagnose and monitor various medical conditions, including liver

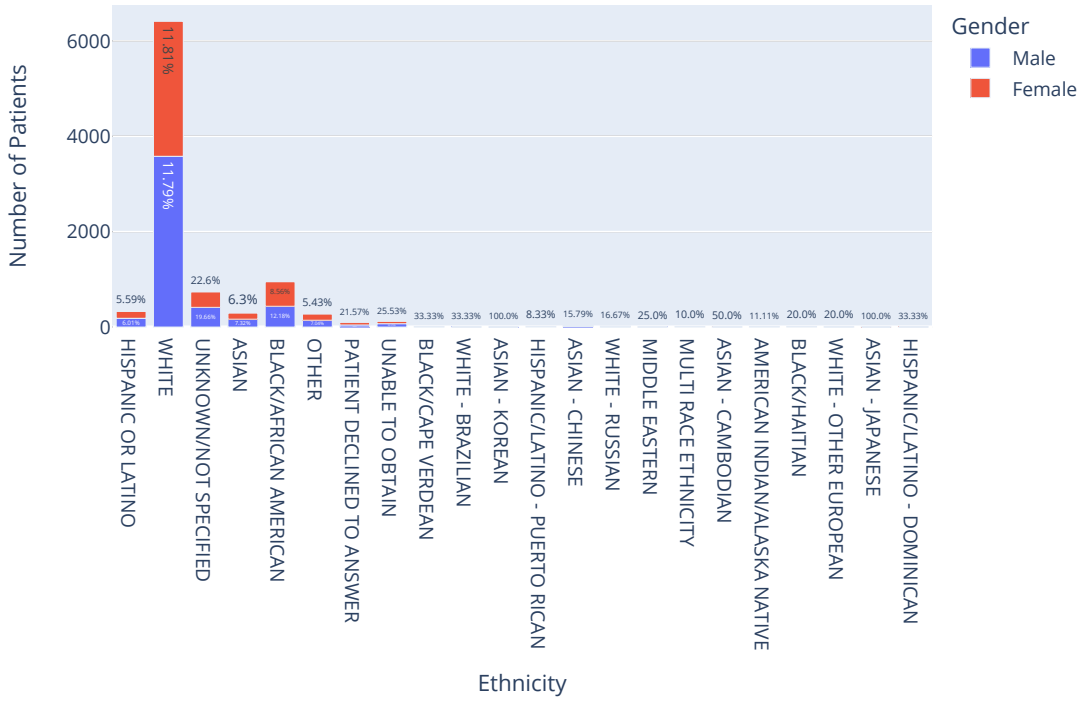


Figure 5.4: Sepsis Mortality by Ethnicity. The percentage inside of each bar represents the mortality % for respective gender & ethnicity

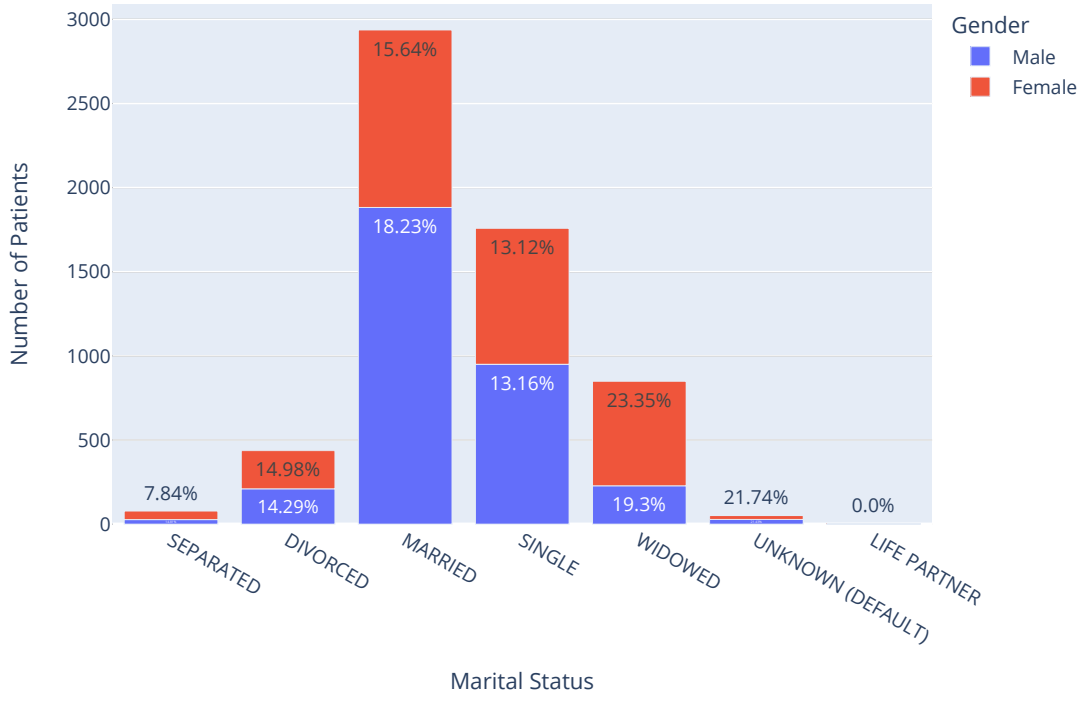


Figure 5.5: Sepsis Mortality by Marital Status

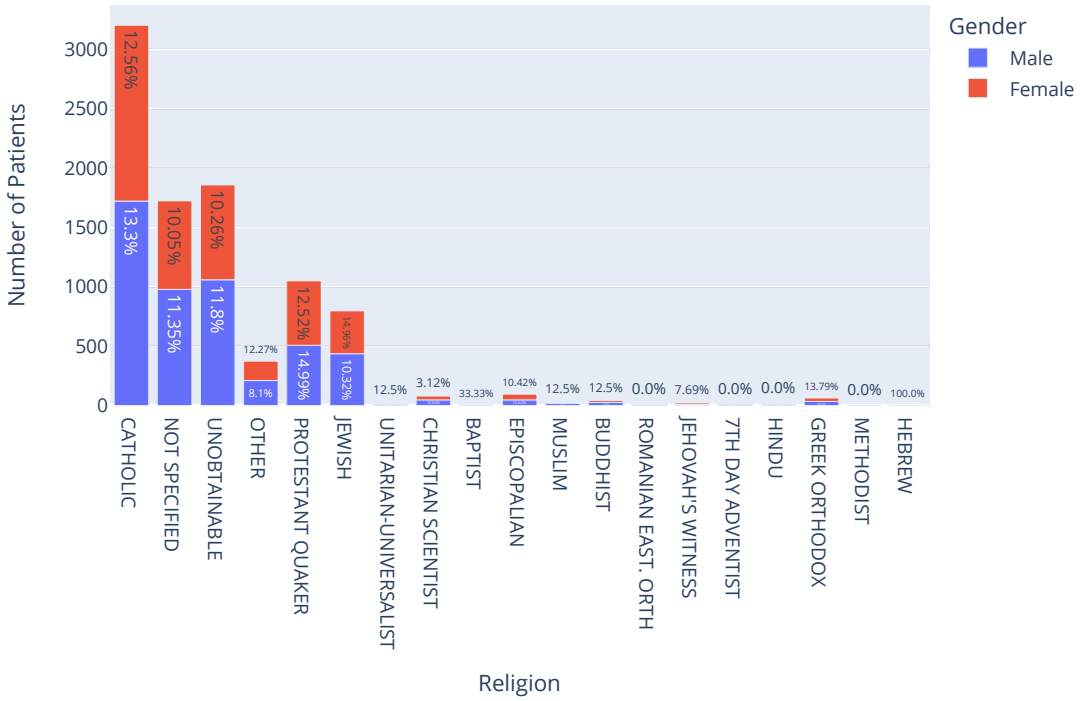


Figure 5.6: Mortality from Sepsis by Religious Group. The percentage inside of each bar represents the mortality % for respective gender & religion

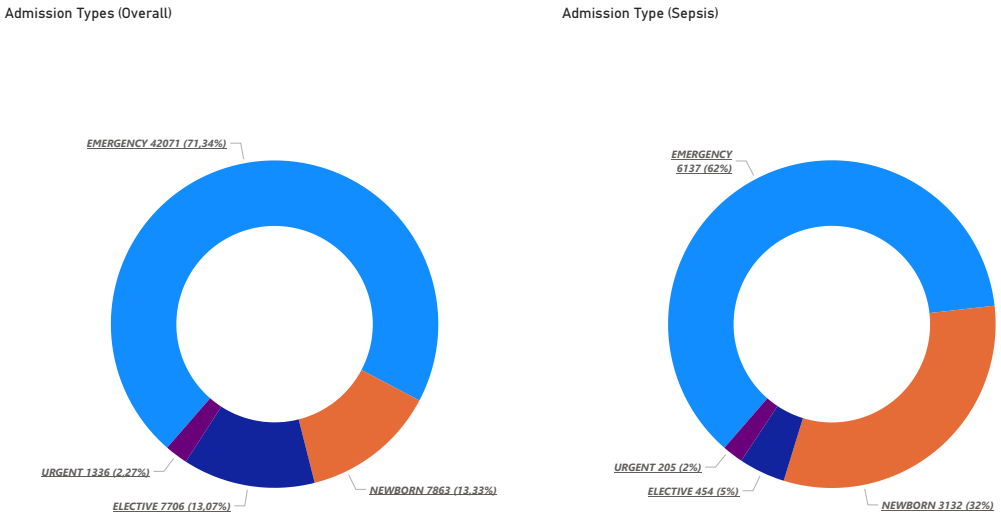


Figure 5.7: Types of admission Overall vs only Sepsis

and kidney disease, diabetes, and electrolyte imbalances. Some standard chemistry tests include:

- Electrolytes: These tests measure the levels of electrolytes in the blood, including sodium, potassium, and calcium.
- Liver function tests: These tests measure the levels of enzymes and other substances produced by the liver and are used to diagnose liver diseases.
- Kidney function tests: These tests measure the levels of substances filtered by the kidneys, such as creatinine and urea, and are used to diagnose kidney diseases.
- Glucose: This test measures the level of glucose(blood sugar) in blood and is used to diagnose and monitor diabetes.

Blood gas tests measure the levels of gases, such as oxygen and carbon dioxide, in the blood. These tests are commonly used to evaluate a patient's respiratory and acid-base balance and to diagnose and monitor conditions such as asthma, pneumonia, and chronic obstructive pulmonary disease (COPD). Some standard blood gas tests include:

- Arterial blood gas (ABG) test: The arterial blood oxygen and carbon dioxide levels are measured to determine the patient's acid-base balance.
- Venous blood gas (VBG) test: This test measures the levels of oxygen and carbon dioxide in the venous blood and is used to evaluate a patient's acid-base balance.
- Pulse oximetry: This test measures the percentage of oxygen in the blood and is used to evaluate a patient's oxygenation.

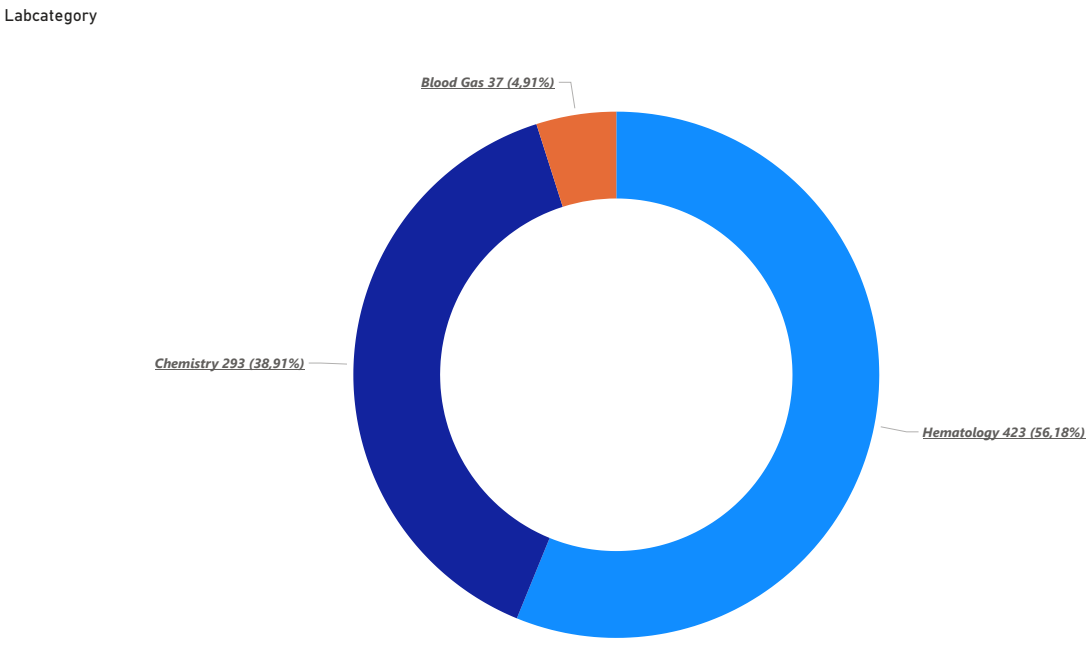


Figure 5.8: Lab category share

The figures 5.9, 5.10 and 5.11 illustrates the procedures performed on the patients diagnosed with sepsis during their admission, the co-diagnoses and the drugs prescribed during their treatment.

Procedures done on patients can be important for predicting mortality in patients diagnosed with sepsis because they can provide information about the severity of the patient's condition and the effectiveness of their treatment. For example, a person with sepsis who needs intensive care or mechanical ventilation probably has a worse condition and a higher chance of dying [RSN+14]. In the same way, let us say a patient with sepsis needs more than one treatment or procedure, like dialysis or surgery. In that case, it may indicate that their condition is not responding well to treatment and their risk of death is higher. Additionally, specific procedures such as source control (removal of the source of infection) can be essential to prevent sepsis from progressing and reduce mortality. Therefore, by taking into account the procedures done on a patient with sepsis, healthcare providers can more accurately assess the patient's risk of death and plan the appropriate course of treatment.

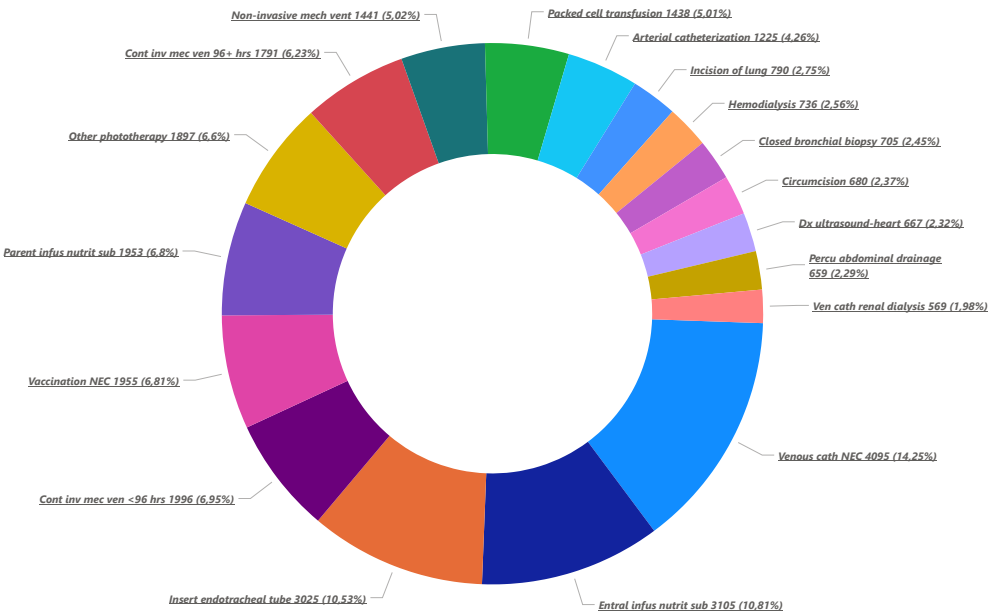


Figure 5.9: Procedures performed on Sepsis Patients

Comorbidities are essential for predicting mortality in patients diagnosed with sepsis because they can increase the risk of complications and death. Comorbidities such as chronic lung disease, heart failure, and diabetes can impair the body's ability to fight off infection, making it harder for the patient to recover from sepsis. Additionally, comorbidities may make it more difficult for healthcare providers to manage the patient's condition, contributing to a higher mortality rate. Therefore, when predicting the mortality of patients with sepsis, it is crucial to take comorbidities into account to more accurately assess the patient's risk and plan the appropriate course of treatment.

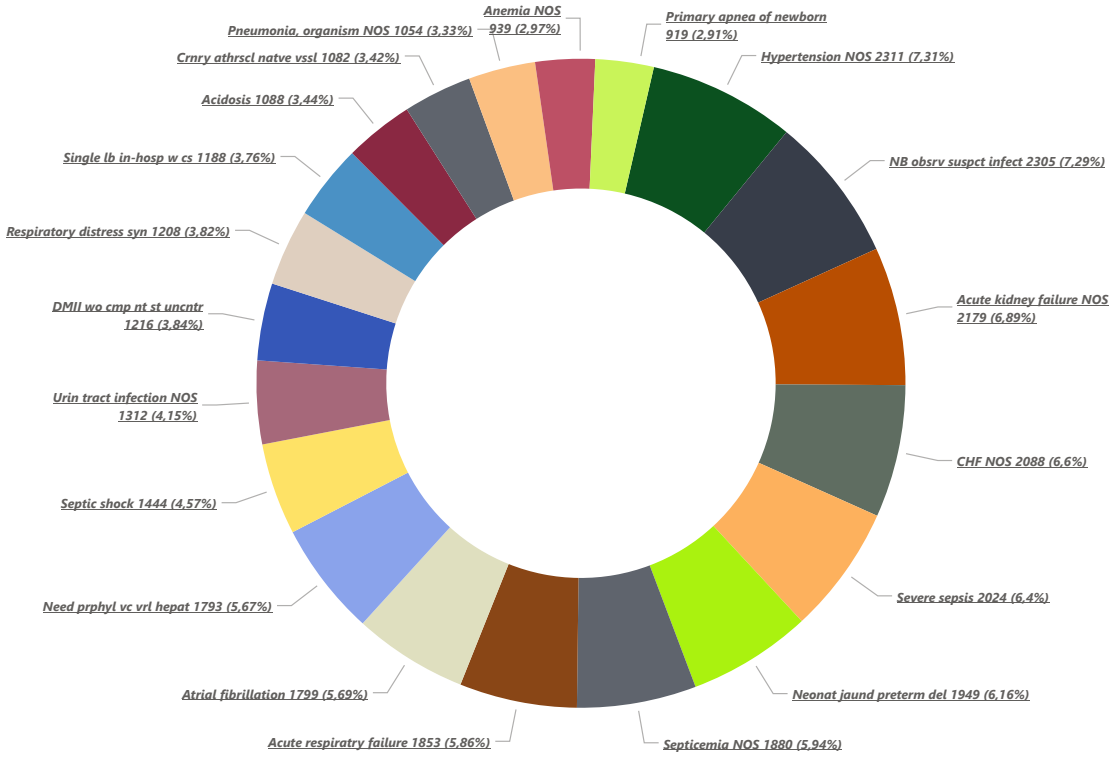


Figure 5.10: Co-Diagnosis of Patients with Sepsis (Top-30)

Medications prescribed to patients can be important for predicting mortality in patients diagnosed with sepsis because they can provide information about the severity of the patient's condition and the effectiveness of their treatment. Medications such as antibiotics, vasopressors, and steroids are commonly used to treat sepsis [SDS+16], and their use can indicate the severity of the patient's condition. For example, if a patient with sepsis requires high-dose vasopressors to maintain their blood pressure, it is likely that their condition is severe and that their risk of death is higher. Similarly, suppose a patient with sepsis is not responding well to antibiotics and requires multiple or high-dose antibiotics. In that case, it may indicate that their condition is not responding well to treatment and their risk of death is higher. Additionally, certain medications, such as activated protein C (APC), have been associated with a reduced mortality rate in sepsis. Therefore, by taking into account the medications prescribed to a patient with sepsis, healthcare providers can more accurately assess the patient's risk of death and plan the appropriate course of treatment.

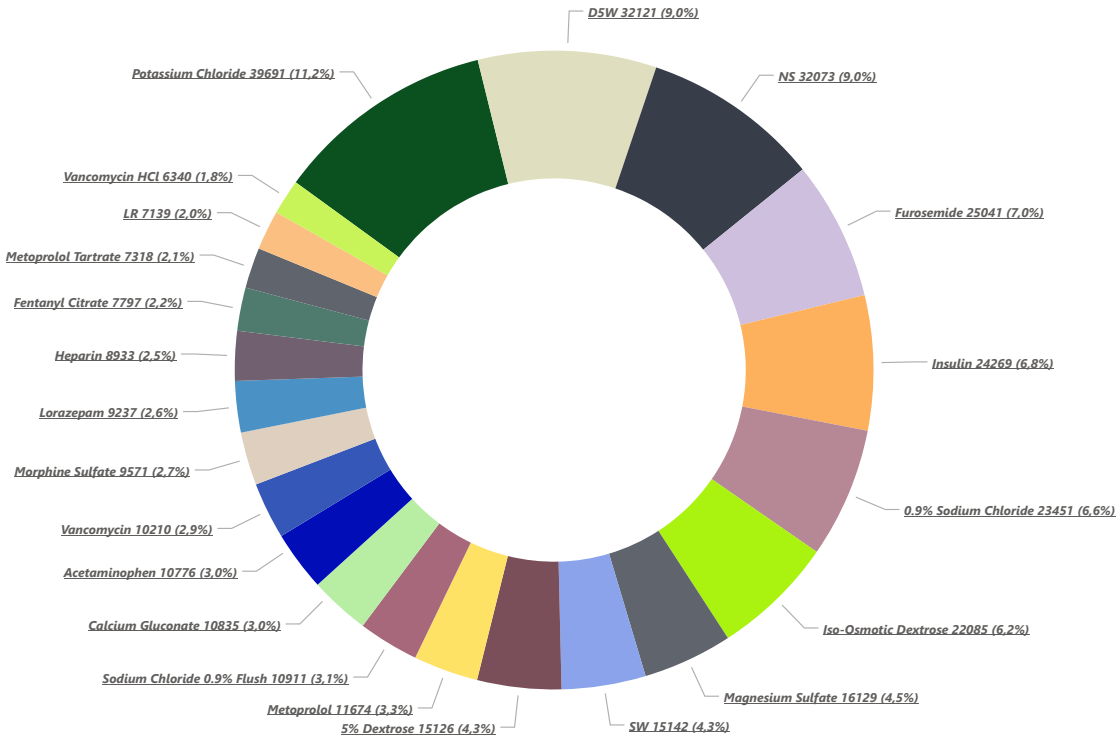


Figure 5.11: Most prescribed drugs to Patients with Sepsis

5.4 Data Preprocessing

Figure 5.12 explains the complete preprocessing carried out on the dataset before passing it to the machine learning model in a flowchart. All the first admissions of patients aged between 18-90 years diagnosed with "Sepsis", "Severe Sepsis", or "Septic shock" are queried from a Neo4j database. These are then filtered to get the latest 24-hours activities (such as completed lab experiments or drugs taken in the last 24 hours of admission). The results of the categories *Labs*, *Vitals* and dosage values in *drugs* involve a few common preprocessing steps.

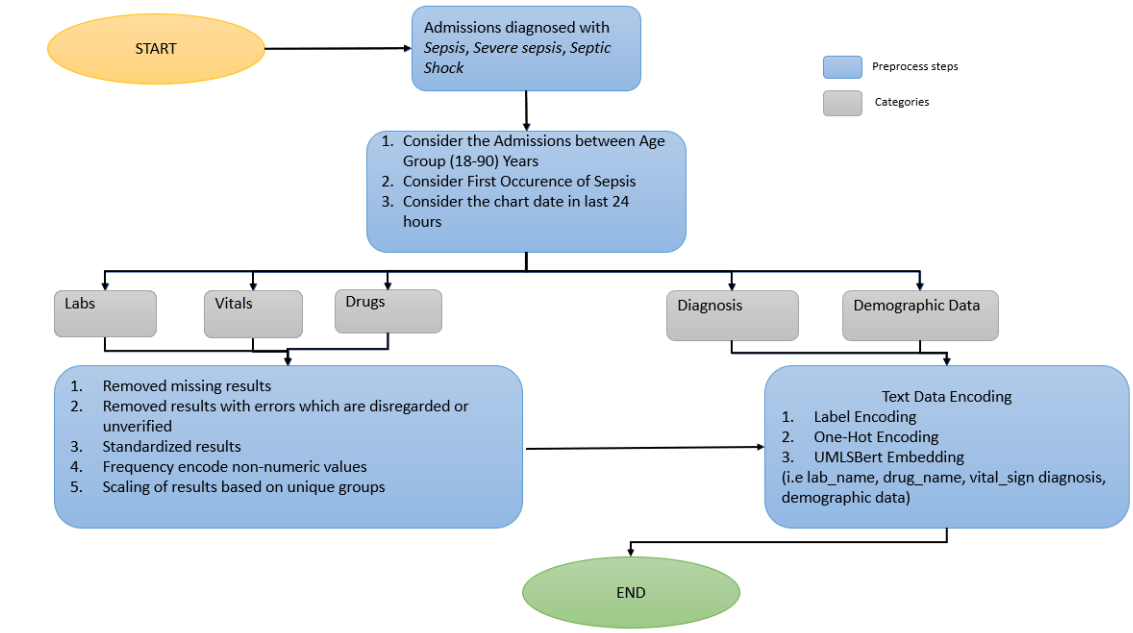


Figure 5.12: Data Preprocessing on MIMIC-III

- Remove missing results or dosage value
- Removed results with error or thode which had an error or were discarded.
- Value formatting, there can be values in Labs results or drugs dosage values such as *>1* or *Greater than 1* which needs to be standardized
- Encoding the categorical data
- Scaling the unique groups, it is done to adjust the features of a dataset so that they are on a similar scale

Different node types (labs, drugs, diagnosis etc.,) include features such as Drug Names, Lab names and vital names. These are categorical in nature but are also free text so applying a text based encoding such as UMLSBert embedding is more beneficial as they capture the context and semantic meaning of words and as discussed in background section 2.8.

The category *Diagnosis* also consists of free text (Diagnosis of the patient). An example of a diagnosis is *Tuberculous pneumonia [any form], tubercle bacilli not found by bacteriological or histological examination, but tuberculosis confirmed by other methods [inoculation of animals].* Thus, we used UMLSBert to encode the different diagnoses.

For the *Demographic Data* categories, such as Gender, Ethnicity, Marital Status and Religion which are categories are encoded using one-hot encodings.

6 Experimental Setup

We provide all the information necessary to reproduce the evaluation results of this research in the following chapter:

- Section 6.1 is devoted to describing experimental input data to the model.
- The Section 6.2 outlines the models used in the experiments and provides implementation details for different steps of the methodology as well as selected hyperparameters.
- The Section 6.3 explains the different metric used for evaluation of performance of the model and evaluate model predictors in this thesis.
- Finally, in Section 6.4 we provide a detailed specification of the hardware and software used.

6.1 Datasets

Throughout the experiments, a heterogeneous graph data object, as shown in the Figure 6.1 is used. The graph describes patients' real-world interaction with the healthcare system. A patient can have multiple admissions for specific(pre-planned surgery) or unspecific reasons(in some emergency); Once admitted, different lab tests can be carried out, drugs might be prescribed, procedures such as *"Transfusion of platelets"* may be performed, vitals get checked and clinical notes get written. The hetero data object formulates this input graph to an object known as a HeteroData object; it is created in Pytorch geometric (PyG) library. To create this object, the following attributes need to be passed.

- Node feature matrix with shape [num_nodes, num_node_features] denoted by **x**. Features on the nodes such as demographic data, encoded lab names, drug names, and diagnosis texts.
- Graph connectivity with shape [2, num_edges] denoted by **edge_index**. Example: from the input graph Figure 6.1, the admission is connected to a lab then indexes of these nodes are passed as a matrix.
- Edge feature matrix with shape [num_edges, num_edge_features] denoted by **edge_attr**. The edges also have weights; in our case, the edge connecting Admission A1 with Lab L1 has lab test values as edge weights, or the Admission A1 connecting to Drug D1 has edge weights, which are drug dosages prescribed by the doctor.

- Labels to train against (may have an arbitrary shape), e.g., node-level targets of shape [num_nodes, num_classes] or graph-level targets of shape [1, num_classes] denoted by **y**. It is used to optimize the algorithm training; in our case, it is the hospital mortality of the patients.

The object 6.1 illustrates the complete data object(balanced) passed to the graph neural network.

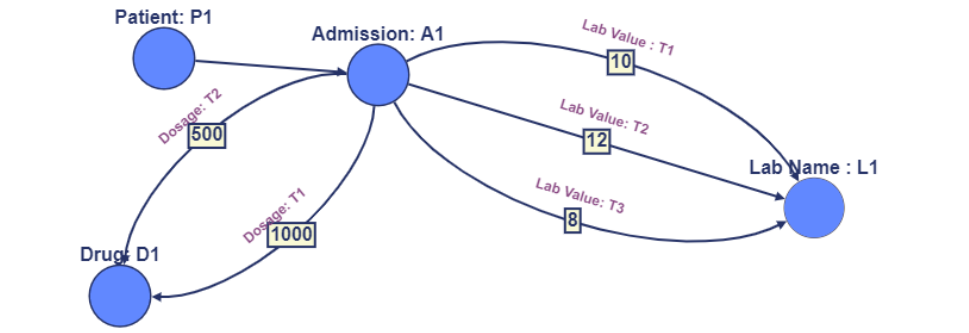


Figure 6.1: Sample input graph

Listing 6.1: HeteroData Object

```
HeteroData(
  num_node_features=3,
  num_classes=2,
  Admission={
    x=[2194, 56],
    y=[2194],
    train_mask=[2194],
    val_mask=[2194],
    test_mask=[2194]
  },
  Labs={ x=[203715, 100] },
  Vitals={ x=[81396, 100] },
  Output={ x=[21988, 100] },
  Drugs={ x=[266064, 100] },
  Diagnosis={ x=[32750, 100] },
  Demography={ x=[86857, 3] },
  (Admission, has_labs, Labs)={
    edge_index=[2, 203715],
    edge_attr=[203715, 1]
  },
  (Labs, rev_has_labs, Admission)={
    edge_index=[2, 203715],
    edge_attr=[203715, 1]
  },
  (Admission, has_vitals, Vitals)={
    edge_index=[2, 81396],
    edge_attr=[81396, 1]
  },
  (Vitals, rev_has_vitals, Admission)={
    edge_index=[2, 81396],
    edge_attr=[81396, 1]
  },
  (Admission, has_ouput, Output)={
    edge_index=[2, 21988],
    edge_attr=[21988, 1]
  },
  (Output, rev_has_ouput, Admission)={
    edge_index=[2, 21988],
```

```
    edge_attr=[21988, 1]
  },
  (Admission, has_drugs, Drugs)={
    edge_index=[2, 266064],
    edge_attr=[266064, 1]
  },
  (Drugs, rev_has_drugs, Admission)={
    edge_index=[2, 266064],
    edge_attr=[266064, 1]
  },
  (Admission, has_diagnosis, Diagnosis)={ edge_index=[2, 32750] },
  (Diagnosis, rev_has_diagnosis, Admission)={ edge_index=[2, 32750] },
  (Admission, has_same_demo, Demography)={ edge_index=[2, 86857] },
  (Demography, rev_same_demo, Admission)={ edge_index=[2, 86857] }
)
```

6.1.1 Data Splitting

A total number of 9928 Patients were identified of whom 8831 (Survived) 1097 (expired). The complete data is used to randomly choose equal number of survived and expired patients. Finally 2194 Patients are taken into account. A stratified (dataset maintains the same proportion of class labels as the original dataset) split of 70% train & 30% test is applied on this dataset using scikit-learn train-test split library. Further, a stratified K-fold cross validation is applied on the training data with K=2. This splits the training data into 2 equal sets (train & validation) on which the GAT gets trained & validated iteratively.

6.1.2 Encoding

The diagnosis text, lab name, drug name, vitals name, and excretion text, which were all available as free text, were encoded into a numerical representation using UMLSBert 2.8. It returns a 768-dimensional vector for each text. It was then passed on to Principal component analysis (PCA) for reducing the dimensionality and achieving computation efficiency. Gender, religion, ethnicity, and marital status, all of which are available as categories, were one-hot encoded 2.8. All the results of lab tests (unique groups), dosage values of drugs (unique groups), and vital signs (unique groups) that are not numerical are frequency encoded 2.8. Frequency encoding is prone to label leakage, thus it is performed individually on each split (train, test and validation).

6.2 Model

A graph attention network, which is explained in detail in the background section 2.7 is used as the primary model to predict the mortality of the patient diagnosed with sepsis. This model is selected over GCN and SAGE model as it allows flexible and efficient computation by incorporating attention mechanism. This allows the model to focus on more important neighbors and to weigh them more heavily in the computation. Another advantage of this model is that it can be applied on varying number of neighbours with different feature dimension on each node whereas both GCN and SAGE assumes that the

graph has fixed number of neighbours for each node. While both GCN and SAGE struggle with scaling, GAT model can efficiently handle large graphs.

6.2.1 Architecture

Figure 6.2 illustrates the model architecture for training a heterogeneous graph. Since GAT, does not support heterogeneous graphs in general, the architecture used in the experiments duplicates the message passing and update function(GATConv) for each individual edge types. The HeteroData object for each node (labs, drugs, diagnosis, vitals) is passed to a GATConv (Graph Attention Convolution). In a GATConv, the flow of computation for an undirected graph can be broken down into the following steps:

1. Linear transformation: The input node features are first transformed using a linear transformation matrix, which is learned during training. This step is typically done with a single-layer perceptron.
2. Attention mechanism: The transformed features are then used to compute an attention coefficient for each neighbouring node of a given node. The attention coefficient is a scalar value that represents the importance of the neighbouring node concerning the given node. This step uses a multi-layer perceptron (MLP) with two linear layers.
3. Attention-weighted aggregation: The attention coefficients are then used to weight the features of the neighbouring nodes, which are then aggregated to form the updated feature vector for a given node.
4. Repeat steps 2 and 3 for all the nodes in the graph.
5. Linear transformation: updated node feature for all nodes is again passed through a linear transformation matrix to obtain a final representation.
6. Activation: This representation is passed through a non-linearity to learn complex non-linear relationships. In case of GATConv leaky relu is used. Finally, it is passed to softmax to convert the output vector to a probability distribution over all classes.

This entire process is done for all the model layers. Each layer's output is passed as input to the next layer.

In a GATConv for heterogeneous graphs, Node type represents different types of nodes such as *admissions, labs or drugs* and Edge type are used to determine which node interacts with which other node. Whereas in a homogeneous graph, all the nodes types interact with each other. Attention coefficients are calculated based on edge types and node types. The attention mechanism in GATConv allows the model to focus automatically on the most relevant nodes and edges in the graph rather than considering all nodes and edges equally.

The pseudo code 2 explains the flow of the code used for different experiments. The complete code can be found in the GitHub repository [2]

[1]https://pytorch-geometric.readthedocs.io/en/latest/tutorial/heterogeneous.html
[2]https://github.com/Aftab571/SepsisMortalityPredictionHetGATConv

Algorithm 2: Pseudo Code

Input: Graph G(V, E); input features x_v, $\forall v \epsilon V$; edge features $edge\_attr_v$, $\forall v \epsilon V$; target condition T

Output: Vector representation z_v, $\forall v \epsilon V$; Edge weight *alpha*; predictions metric AUC

for $Adm\ in\ filter(Diagnosis = T)$ **do**
    df_labs $\leftarrow preprocess(getLabs(Adm))$
    df_drugs $\leftarrow preprocess(getDrugs(Adm))$
    df_vitals $\leftarrow preprocess(getVitals(Adm))$
    df_CoDiagnosis $\leftarrow preprocess(getCoDiagnosis(Adm))$
end
df_final $\leftarrow equalize(Adm[Survived, Dead])$
train_mask, val_mask, test_mask $\leftarrow create\_train\_val\_test\_split(df\_final)$
heteroObj $\leftarrow createHeteroObj(train\_mask, val\_mask, test\_mask)$
$Initialize\ GATConv(in\_channels, out\_channels, attention\_head, edge\_attr_{dim})$
for $i\ in\ epoch(0, 300)$ **do**
    $out_i, alpha_i, loss_i \leftarrow train(GATConv(heteroObj)[train\_mask])$
    $optimize(loss_i)\ plot(loss_i)$
end
acc_{val}, $preditions_{val}$ ←eval(GATConv(heteroObj)[val_mask])
$acc_{test}, preditions_{test}$ ←eval(GATConv(heteroObj)[test_mask])
$AUROC$ ←report_eval_stats(predictions$_{test}$)
for $x\ in\ [labs, drugs, diagnosis, vitals]$ **do**
    $alpha_x$ ←map(alpha, train_mask, x)
end
z_v ←map(out['Adm'][train_mask])
$Important\_features$ ←plot(sort[alpha$_x$]$desc$)

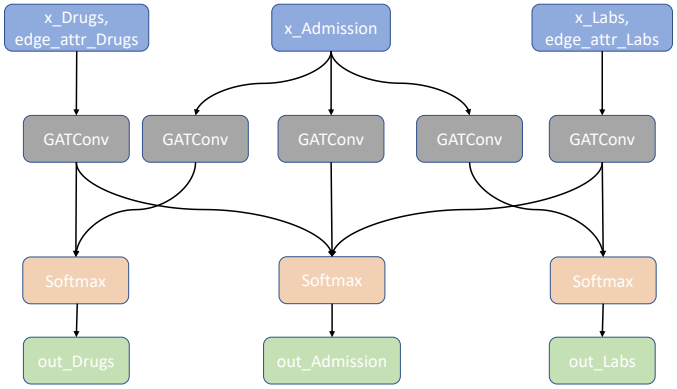


Figure 6.2: GAT Architecture for multiple entities (undirected) adopted from PyG [1]

6.2.2 Hyper-parameters

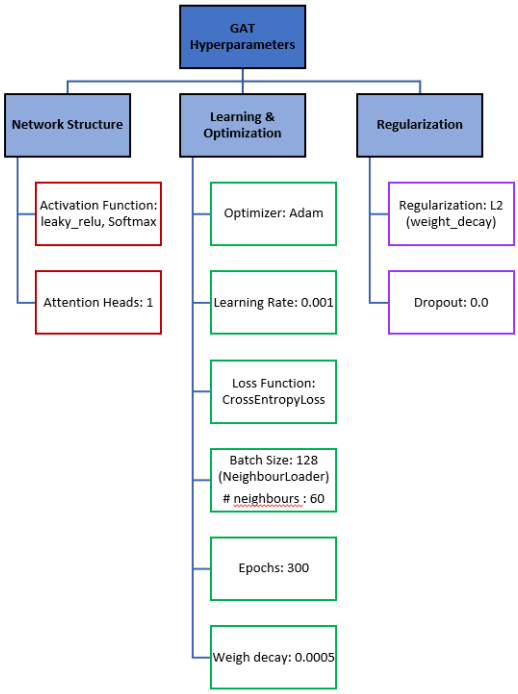


Figure 6.3: GAT Hyperparameters

6.3 Metric

This section explains the interpretation of each metric used in the evaluation of this thesis.

- **Sensitivity:** it is also known as the "true positive rate" or "recall," is a measure of the proportion of true positive predictions made by a binary classification model

out of all the actual positive instances. It is the ratio of true positive predictions to the total number of actual positive instances.

Sensitivity = (True Positives) / (True Positives + False Negatives)

In the case of this thesis a positive class denotes Survival of the patient. A model with high sensitivity can correctly identify a large number of positive cases. It is crucial when the cost of false negatives (missed detections), like in medical diagnosis or fraud detection, is high.

- **Specificity:** it is alternatively known as the "true negative rate," is a measure of the proportion of true negative predictions made by a binary classification model out of all the actual negative instances. It is the ratio of true negative predictions to the total number of actual negative instances.

  Specificity = (True Negatives) / (True Negatives + False Positives)

  In the context of this thesis, a negative class denotes a patient's death. A model with high specificity can correctly identify a large number of negative situations. It is important in situations where the cost of wrong detection, or "false positives," is high, like in medical diagnosis or security systems.

- **AUROC:** AUROC stands for "Area Under the Receiver Operating Characteristic Curve." Receiver Operating Characteristics (ROC) is a graph depicting the performance of a binary classifier system when the discriminating threshold is altered. The ROC curve is derived by graphing the true positive rate (TPR) vs the false positive rate (FPR) at different threshold levels. The area under the ROC curve (AUROC) evaluates the classifier's ability to differentiate between the two classes. A perfect classifier would have an AUROC of 1, whereas a random guessing classifier would have an AUROC of 0.5. AUROC is a popular measure because it is insensitive to the choice of the classifier's threshold, and it is a measure of the model's overall performance.

  It is important to note that a high-sensitivity model doesn't always mean a high-specificity model, and vice versa. A trade-off is often made between sensitivity and specificity, depending on the particular use case. It is reason why AUROC is favoured than accuracy of the model.

- **Percentile rank:** it is also known as relative standing, is a way of describing the position of a value within a set of data. It is expressed as a percentage of values equal to or less than the value in question. For example, if a value has a percentile rank of 80, it means that 80% of the values in the set of data are equal to or less than that value.

  $$Percentilerank = \frac{\text{Number of values below the value}}{\text{Total number of values}} x100$$

  The percentile rank is often used in statistics and data analysis to compare the relative standing of a value to the rest in the data set. It can be used to compare

Library	*Version*	*Library*	*Version*	*Library*	*Version*	*Library*	*Version*
comet-ml	**3.31.12**	NumPy	**1.23.1**	tokenizers	**0.12.1**	torch-sparse	**0.6.14**
matplotlib	**3.5.2**	pandas	**1.4.3**	torch	**1.12.0**	torch-spline-conv	**1.2.1**
neo4j	**4.4.5**	plotly	**5.9.0**	torch-cluster	**1.6.0**	torchaudio	**0.12.0**
nltk	**3.7**	scikit-learn	**1.1.1**	torch-geometric	**2.0.4**	torchmetrics	**0.9.3**
NumPy	**1.23.1**	sentence-transformers	**2.2.2**	torch-scatter	**2.0.9**	torchvision	**0.13.0**
tqdm	**4.64.0**	transformers	**4.21.0**	xgboost	**1.6.2**		

Table 6.1: Experiment setup Library information

different groups of data, for example, to compare test scores of students or to determine how an individual's score compares to that of a group of people. In the context of this thesis, percentile rank is used to determine the relative standing of the mortality predictors.

6.4 Experimental Environment

In this section, we overview the essential details about the experimental setup we deploy, running the experiments of this thesis. We employed the following configurations:

Machine Configuration - Database

- **Operating System** Ubuntu 18.04.6 LTS
- **Processor** 4× Intel(R) Xeon(R) CPU E5-2609 v2 @ 2.50GHz
- **Memory** 256 GB RAM
- **Database** Neo4j Kernel (Community Version : 4.4.5)

Machine Configuration - Program Execution

- **Operating System** Ubuntu 18.04.6 LTS
- **Processor** 4×Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz
- **GPU** NVIDIA GeForce RTX 2080 Ti
- **Memory** 32GB RAM
- **CUDA** 10.2
- **Programming Languages** Python (Version 3.8.10)
- **Programming Tools** Visual studio code, Jupyter Notebook (Version 6.4.8)
- **Graph framework** PyG (PyTorch Geometric)
- **Libraries** Refer table 6.1

7 Results & Evaluation

In this chapter, the evaluation results are presented and discussed for the experiments conducted to address the research questions raised in Section 5.1. This chapter is outlined as follows:

1. In Section 7.1, the effect of different representations of data in the graph (graph modelling) on the model performance and their GPU usage and processing time is reported and these results are evaluated to understand their usage.
2. Section 7.2 reports the GAT models performance and the top predictors of the mortality. These predictors are evaluated on their percentile ranking with predictors of SAPS-II and SAPS-III models.
3. In Section 7.3, the effect of different relationship types on GAT is reported and the nature of bias and correlation of the combinations are evaluated.

7.1 RQ-1 Graph data modelling

Modeling the graph data is a process of representing data in the form of nodes & edges and assigning their properties. Determining whether a property should be stored in a node or edge is based on the property and its relationship with other nodes in the graph. This research question is an ablation study of graph representation. Here, three different representations shown in figure 7.1, 7.2 and 7.3 are experimented with GAT to analyze their effect on the performance of the model. All the representations explained below follow a common initial structure wherein the patients can be admitted multiple times to the hospital, and on each admission, different labs and procedures are carried out, vital signs are measured, and drugs are prescribed. The activities performed on the patients can be carried out many times at different time intervals during their stay in the hospital.

- Representation-1 (Figure 7.1): In this representation, every single activity is stored as an edge, with the timestamp and the measured value as properties of the edge. The edge is then mapped to the activity name.
- Representation-2 (Figure 7.2) is similar to the first representation but instead of having multiple edges for the same activity, it has one, which aggregates (SUM, MEAN, MIN or MAX) all activities.
- Representation-3 (Figure 7.3) is similar to the first representation but all the properties present on the edges are transformed into properties of the node. Therefore, each activity only has one edge between each admission and lab.

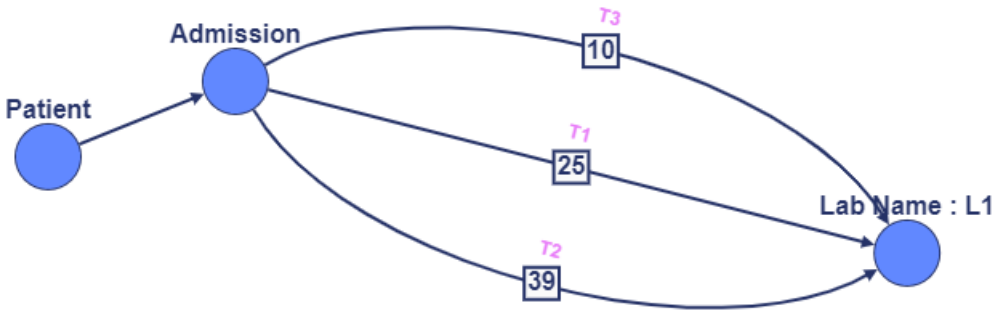


Figure 7.1: RQ1: General Representation

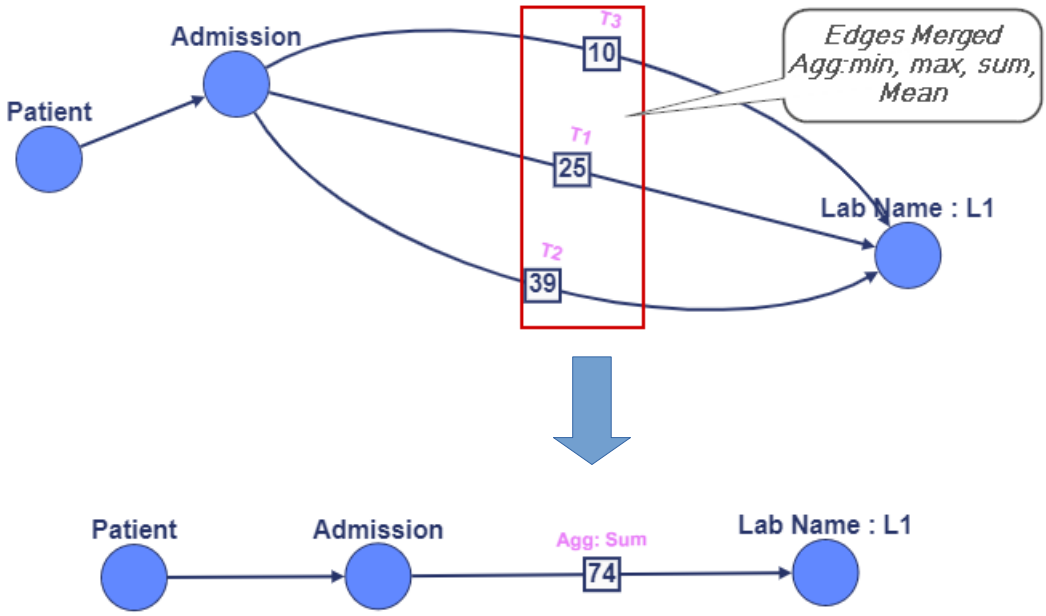


Figure 7.2: RQ1: Edge Merge Representation

Representation	Number of Node Features	Number of edge Features	Number of Edges
General - (Representation-1)	768	1	311245
Mean	768	1	113335
Sum	768	1	113335
Min	768	1	113335
Max	768	1	113335
Edge Data on Node	795	0	113335

Table 7.1: Number of Node, Edge features and Total number of edges

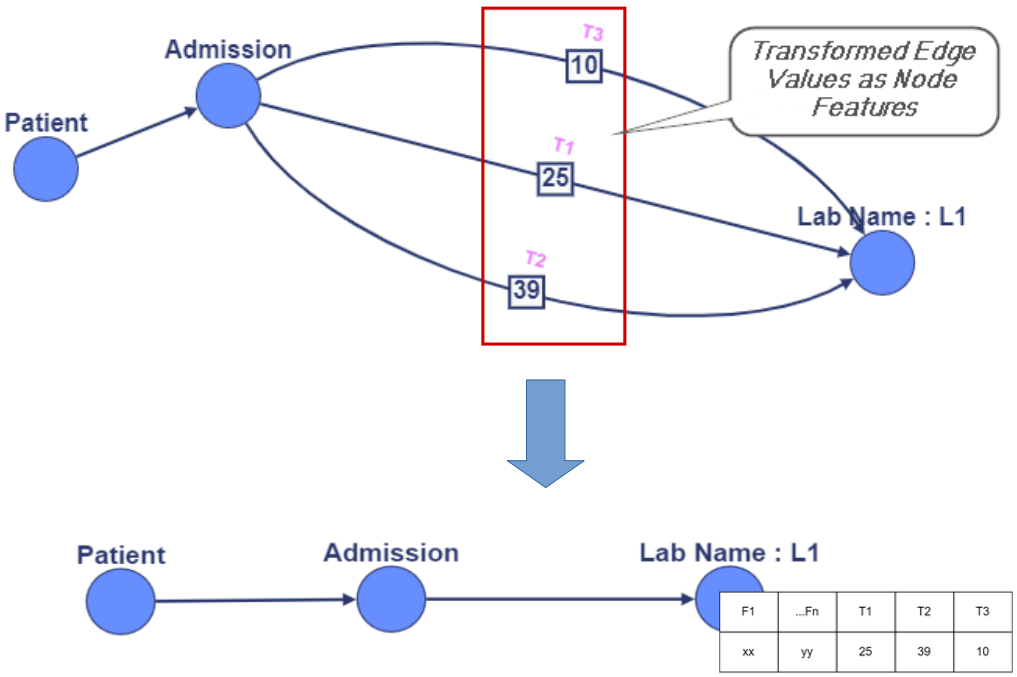


Figure 7.3: RQ1: Edge data on Node

All three representations are then passed to the GAT to predict the mortality of each patient. Table 7.1 shows the total number of features present on nodes and edges and also the number of edges for each representation that are passed to the model and Table 7.2 shows the results from the model. The model's performance is measured with the metrics Sensitivity, Specificity, and AUROC. It can be seen from the results that all representations have nearly the same performance (64.0% - 66.0%). Significant differences occur in the memory and time consumption for running the model. Figures 7.4, 7.5, and 7.6 show how different representations used GPU and their processing times.

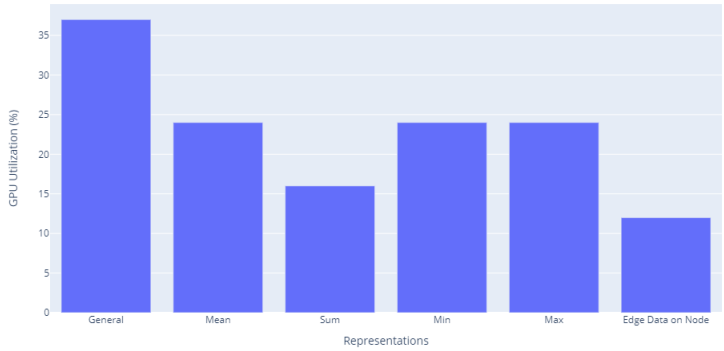


Figure 7.4: GPU utilization during model training on different representations

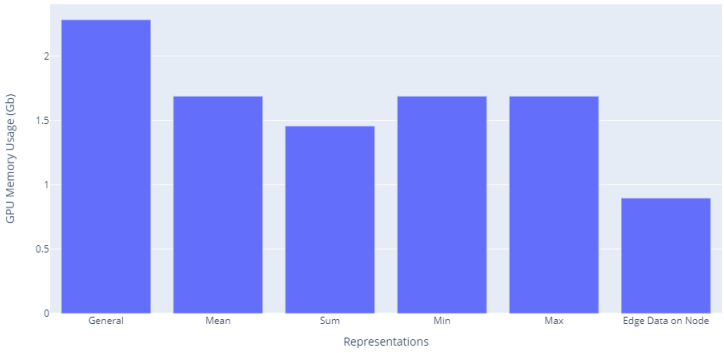


Figure 7.5: GPU memory usage during model training on different representations

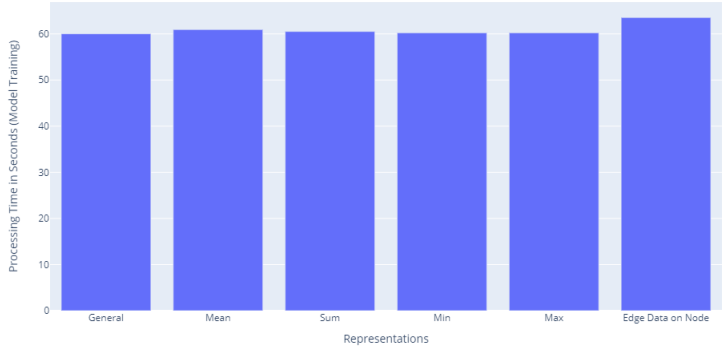


Figure 7.6: Time taken during model training on different representations

The memory and time consumption of aggregated edge features representation and edge features as node features is comparatively lower than the general representation (Representation-1) for the same AUROC of the model. The reason for this is less number of edges between the nodes (63.5% fewer edges when compared to general representation) as can be seen in table 7.1. However, there are a few problems to be taken into account before going for any representation.

Table 7.2: Results of different representations of the graph

Representation Type	Aggregation Type	Specificity	Sensitivity	AUROC
General	**-**	**0.692**	**0.63**	**0.66**
Edge Merged	***Mean***	**0.719**	**0.586**	**0.66**
	Sum	**0.625**	**0.674**	**0.649**
	Min	**0.637**	**0.659**	**0.648**
	Max	**0.658**	**0.629**	**0.643**
Edge data on Node	-	**0.664**	**0.617**	**0.64**

- In Edge Merged representation (representation-2), the edges are aggregated. It introduces a bias in the model. For example, certain lab tests were conducted more often than others. In this case, an aggregation such as sum gets a higher value which in turn, during the matrix factorization of the GAT model will allocate more weight to these labs. Whereas the aggregation function such as min or max can act as better predictors of mortality of the patients as it can be seen in 2.1 and 2.2. The worst values for the last 24 hours are used while calculating the SAPS-III score. In order to take the worst value there needs to be a baseline to compare them which requires domain knowledge. The usage of min or max aggregators without any domain knowledge of all the lab values induces a negative bias in the prediction. Adding domain knowledge to this representation can solve the bias problem.

- In Edge data on Node representation(representation-3), one advantage is that lab values taken at different timestamps are preserved as a new feature on the node. However, there can exist certain labs which were less frequently carried out, leading to missing values in the feature matrix. For example, in figure 7.7, Lab-2 for the patient was only done twice, whereas, for lab-1, it was done thrice. These missing value needs to be somehow masked. This induces a bias in the model's prediction, affecting the model's ability to capture the underlying pattern in the data. The missing values were masked with zeros in this representation. An extension in the PyTorch library to handle variable-size tensors can solve this issue.

Patient	F1...Fn	T1	T2	T3
Lab -1	xxx	12.0	13.0	9.8
Lab -2	yyy	133	145	-

Figure 7.7: Missing data in Representation-3

Encoding	Sensitivity	Specificity	AUROC
Label- encoding	0.95	0.052	0.501
One hot encoding	0.779	0.814	0.797
UMLSBert Encoding	**0.888**	**0.847**	**0.867**

Table 7.3: Different encodings on Drugs

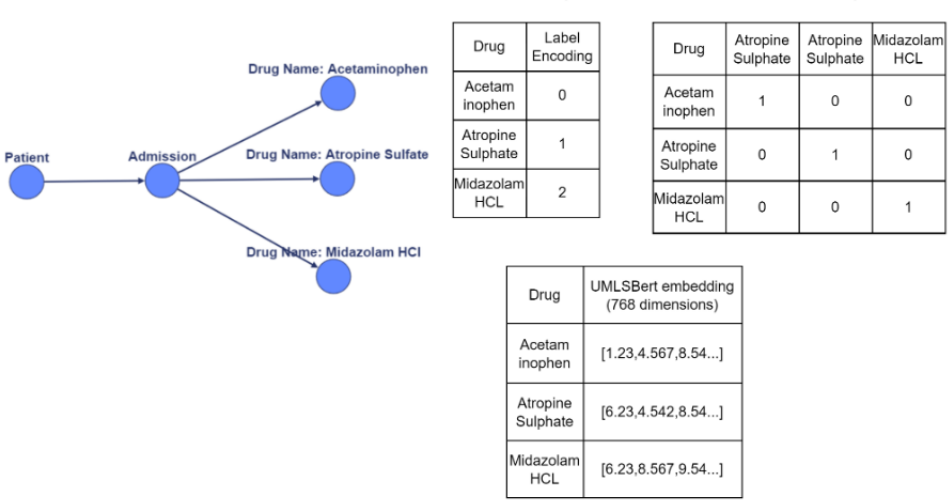


Figure 7.8: Visualization of different encoding

One of the sub questions answered relevant for the representation is the impact of different feature encodings discussed in 6.1.2. Encoding the categorical or text data into any numerical form is necessary as machine learning models can only use numerical data. The figure 7.8 can be used to visualize how the encoding looks before passing them to the model. Table 7.3 explains how different embeddings impact the predictive power of the model. The encoding with UMLSBert (AUROC: 86.7%) clearly outperforms both Label encoding (AUROC: 50.1%) and one-hot encoding (AUROC: 79.7%) as the UMLSBert is fine-tuned on a large corpus of medical texts and has been trained to understand medical concepts, terminology and context. However, to generate text embeddings from a UMLSBert pre-tuned model additional time is required in the preprocessing step and also the quality of the embedding depends on the underlying data used for pre-training and fine-tuning. The one-hot encoding on the other hand does not require such a pre-trained model but still performs better than label encoding as the relationships between the categories are not misinterpreted as a numerical ordering.

7.2 RQ-2: Mortality prediction and evaluation of predictors

In this question, the GAT model using the architecture explained in section Figure 6.2 of evaluation setup and using the hetero graph object Listing 6.1 is evaluated for predicting the mortality of patients diagnosed with sepsis. The model is evaluated on 30% (659) test data of the total 2194 patients. The train, test & validation accuracy and the training

loss of the GAT model are provided in Figure 7.9 and Figure 7.10 respectively. They provide information about how well the model is learning over 300 epochs and is able to generalize on unseen test data. The confusion matrix Figure 7.11 shows the distribution of correctly and falsely predicted values on the test data. Table 7.4 shows all measures derived from the confusion matrix. The model reports an 87.5% AUROC with a specificity of 83% and sensitivity of 91.5%. The measures indicate that the model distinguishes between survived and dead patients fairly. Furthermore, we evaluated which features (predictors) are important in the model for prediction. Therefore, we retrieved all edges after the model was trained. Then, we determined the target of each edge (lab, drug or diagnosis). Then the edge weights for unique lab tests are summed up to calculate their percentile rank (i.e., their importance). The figure 7.13 shows the 30 most important (highest percentile ranks) predictors in the relationship between Admissions and Labs. To compare the goodness of these important features they are compared against the well-established SAPS-II and SAPS-III models. Figure 7.12 shows all the lab-related features which are used as important predictors for mortality in SAPS-II and SAPS-III are ranked in the top 90 percentile (out of 468 features) with the majority being in the 95 percentile in features that are important predictors in the GAT model. This shows that the GAT model emphasizes on the differentiating features of the mortality while learning the underlying pattern. Similarly, the same feature correspondence can also be seen in the relationship between the Admissions and vital signs in the Figure 7.14. All the plotted vitals are used as predictors in both SAPS-II & SAPS-III scores.

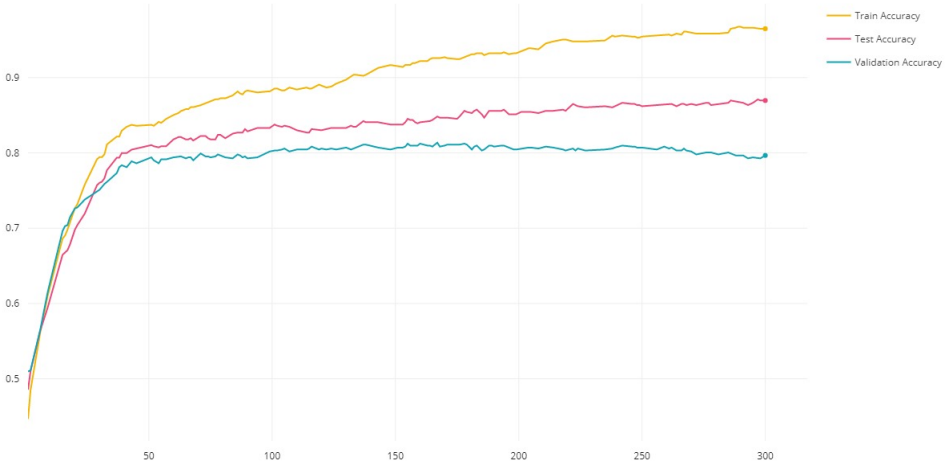


Figure 7.9: RQ2: Train, Validation & Test Accuracy for GAT

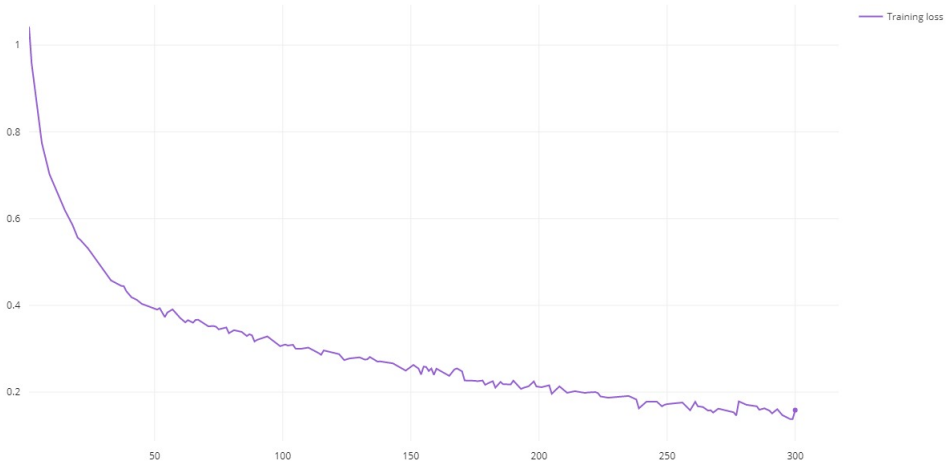


Figure 7.10: RQ2: Training loss

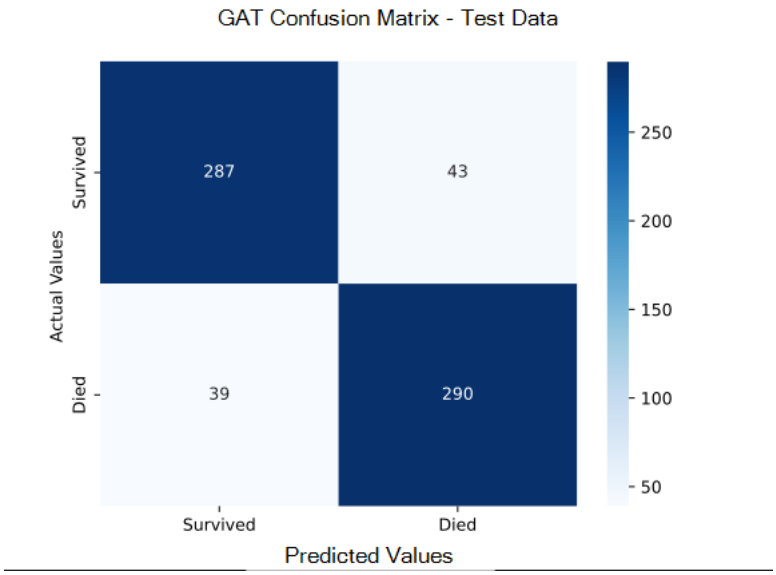


Figure 7.11: RQ2: GAT Confusion matrix on Test Data

Measure	Value
Sensitivity	**0.9150**
Specificity	**0.8329**
Area Under the Receiver Operating Characteristic	**0.8756**

Table 7.4: Measures derived from confusion matrix for the GAT models prediction

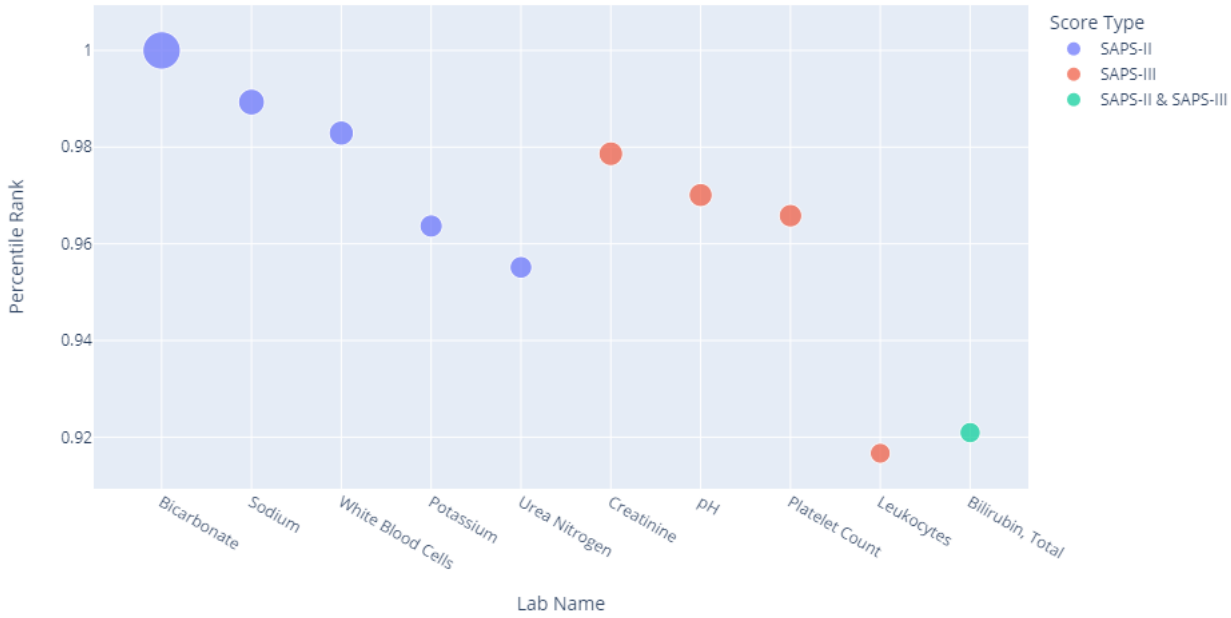


Figure 7.12: RQ2: GAT edge ranking for Labs comparing with SAPS-II & SAPS-III features

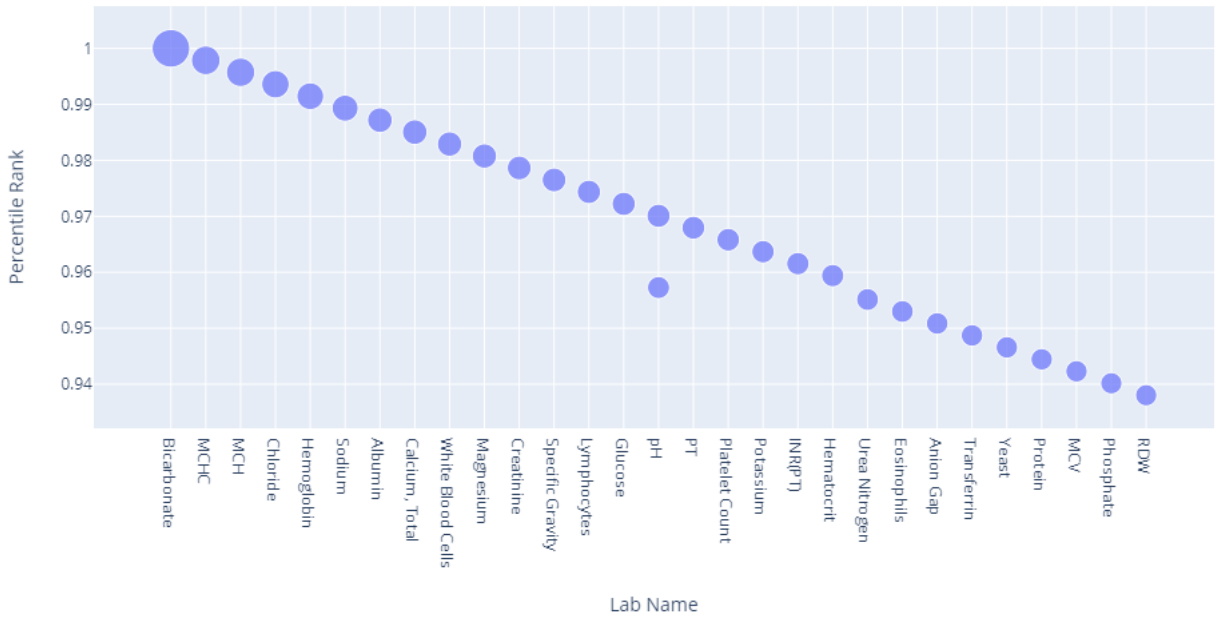


Figure 7.13: RQ2: GAT Top-30 features

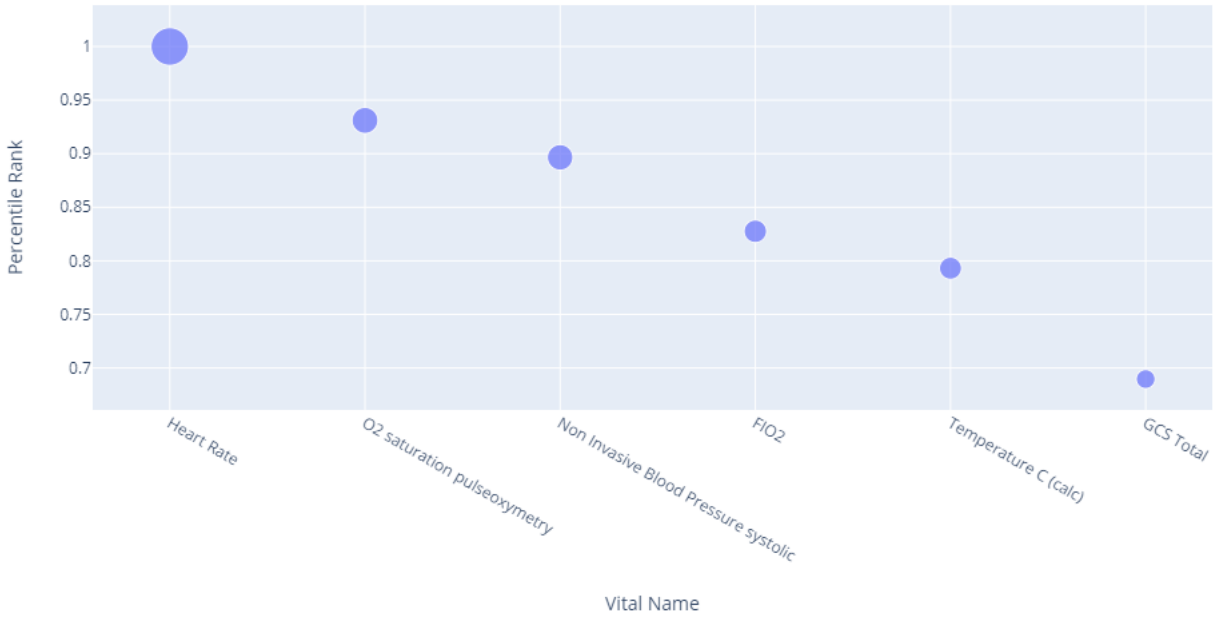


Figure 7.14: RQ2: GAT edge ranking for Vitals comparing with SAPS-II & SAPS-III features

7.3 RQ-3: Effect of different combinations of relationship types & the nature of bias

This question answers how important different relationship types are in the mortality prediction of patients diagnosed with Sepsis. Table 7.5 shows the effect of each individual relationship type with single node types in predicting the mortality of the patients. The node type drugs (AUROC: 85.8%) act as better predictors than labs(AUROC: 66.0%), diagnoses(AUROC: 81.7%), or vitals(AUROC: 71.3%) because the prescription of drugs is based on a diagnoses, which involves some information about the patient's underlying condition. This is a disadvantage, but it can be used under certain circumstances, such as to predict the mortality of patients by trying different permutations & combinations of drugs before administering them to the patients. Furthermore, it can also be used to check which drugs have a higher mortality risk. Additionally, the use of drugs as predictors introduces some bias as they are heavily dependent on the labs and vitals of the patients.

Relationship Type	Sensitivity	Specificity	AUROC
Admission - Drugs	**0.841**	**0.876**	**0.858**
Admission - diagnoses	0.819	0.816	0.817
Admission - Vitals	0.730	0.694	0.713
Admission - Labs	0.63	0.692	0.66

Table 7.5: Effect of individual relationship type with single node types on mortality

The tables 7.6, 7.7, 7.8, & 7.9 shows the effect of different combinations of node types. what follows after the admission of a patient (i.e. a patient gets admitted, then vital signs are checked, then lab tests are done, followed by diagnoses of the patient leading to the drug prescription). This sequential treatment procedures leads to more information about each patient within this process. It is observed that a sequential correlation of AUROC exists with respect to patients' sequential treatment procedures. Consequently, considering the combination of node types labs and diagnoses (AUROC: 74.1%) or labs and drugs (AUROC: 79.7%) improves the predictive power of the model compared to considering only the node type labs (AUROC: 66.0%). Similarly, there is also an increase for the AUROC when considering the combination of node types vitals and diagnoses (AUROC: 77.1%) or vitals and drugs (AUROC: 81.9%) compared to considering only the node type vitals (AUROC: 71.3%). However, there is a single exception for this pattern (combination of the node types vitals and labs). One reason for this might be that they are independent of each other (i.e a lab is not done based on vital signs but more based on symptoms) and, therefore, do not contribute further information to each other.

In a real-world scenario, diagnoses information and drug prescriptions are usually accessible in the later stages of admission. Therefore, labs and vitals have a much higher relevance for predicting mortality in the early stages of admission.

Relationship Type	Additional Relationship	Sensitivity	Specificity	AUROC
Admission - Labs	*Drugs*	**0.805**	**0.789**	**0.797**
	diagnoses	0.699	0.783	0.741
	Vitals	0.580	0.759	0.669

Table 7.6: Effect of additional relationship in combination with Admission - Labs

Relationship Type	Additional Relationship	Sensitivity	Specificity	AUROC
Admission - Drugs	*Vitals*	0.761	0.876	0.8191
	diagnoses	**0.820**	**0.858**	**0.839**

Table 7.7: Effect of additional relationship in combination with Admission - Drugs

Relationship Type	Additional Relationship	Sensitivity	Specificity	AUROC
Admission - diagnoses	*Vitals*	0.752	0.790	0.771

Table 7.8: Effect of additional relationship in combination with Admission - diagnoses

Relationship Type	Additional Relationship	Sensitivity	Specificity	AUROC
Admission - Labs - Drugs	*diagnoses*	0.781	0.856	0.819
	Vitals	**0.808**	**0.856**	**0.832**

Table 7.9: Effect of additional relationship in combination with Admission - Labs - Drugs

Unlike multi layer perceptrons (MLP), a graph neural network takes into account the relational information between the nodes, which makes them powerful algorithms for handling graph-structured data. But does it mean that adding more relationships gives better predictive power? In order to answer this question, a new relationship type "Demography" is created based on the demographic data **GAMER**(**G**ender, **A**ge-group, **M**arital status, **E**thnicity and **R**eligious status) between the admissions of the patients. The Figure 7.15 depicts the addition of a *same demography* relationship between the admission nodes. It is crucial to note that the relationship created is highly biased towards certain age-group, ethnicity, marital & religious status as discussed in the section 5.3 of the design chapter. The addition of this relationship should highly influence the predictions of the model towards mortality of the patients which can be seen in the results of ***Admission - Demography*** in the table 7.10.

Relationship Type	Sensitivity	Specificity	AUROC
Admission - Demography	**0.10**	**0.919**	**0.512**

Table 7.10: Biased relationship

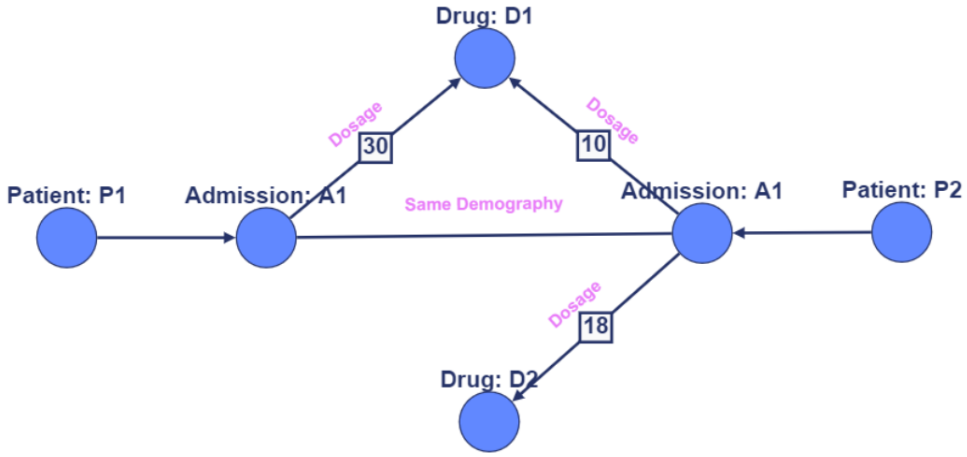


Figure 7.15: Addition of same demography relationship between the admissions

Now when this relationship is added along with other edge types, it can be seen in table 7.11 that this new relationship has only a small influence on the predictions of the model. The reason for this small influence is that the GAT's attention mechanism is designed to calculate attention coefficients, and these coefficients calculated on biased edges are less impactful.

Relationship Type	Additional Relationship	Sensitivity	Specificity	AUROC
Admission - Labs	**Demography**	**0.638**	**0.631**	**0.634**
Admission - Drugs		**0.774**	**0.819**	**0.797**
Admission - diagnoses		**0.771**	**0.784**	**0.777**
Admission - Vitals		**0.796**	**0.490**	**0.643**

Table 7.11: Effect of Demography relationship with other relationship types

Thus it can be concluded that for the GAT model, the impact of bias depends on the kind of bias introduced. The nature of bias can come from different edge weights, missing edges, or unbalanced edges (imbalance in the number of edges for different node types).

8 Conclusion and Future Work

This chapter summarizes the work in this thesis and presents the conclusions obtained from this work. It also proposes some future practices which may extend this research. This chapter is structured as follows:

- In Section 8.1, important conclusions derived from our thesis work is presented.
- In Section 8.2, some extensions to the existing work are presented as future work.

8.1 Conclusion

The research presented in this thesis involved the construction of different graph representations from the MIMIC-III dataset. It investigated the impact of these representations in the graph on the GAT model. Three different types of representations with Admission - Lab's relationship were experimented with. It was found that all representations performed comparatively similar on the AUROC metric (64.0% - 66.0%). Nevertheless, the representation in which edge data was transformed as node features performed best on GPU utilization (12%) and GPU memory usage (0.897 Gb). Although all the representations can be utilized, it can be seen that the representation with edge aggregation induced a bias without domain knowledge and the representation with edge data transformation as node features induced bias by masking missing values. Though the GAT model theoretically allows variable length node features but the current limitation of passing fixed shaped tensor in PyTorch limits the possibility of handling missing values. Thus general representation (Representation-1) (Figure 7.1) was selected. Furthermore, the thesis explained the effects of different encodings on the performance of the model. The predictive power of the model increased by 36.6% with UMLSBert and 29.6% with one-hot encoding of features when compared to label encoding.

Using this general representation with all possible relationships and node types, a GAT model was trained. The model reports an 87% AUROC with 91.5% Sensitivity and 83.3% Specificity. The goodness of the model was quantified by comparing the percentile ranking of edge weights of top predictors of the trained GAT model with the predictors of the SAPS-II & SAPS-III model. It was found that lab predictors (bicarbonate, sodium, potassium, white blood cells, urea nitrogen, bilirubin, creatinine, pH, platelet count, and leukocytes) in SAPS-II & SAPS-III are ranked amongst the top 90-95 percentile. Vital signs predictors (heart rate, O_2 saturation pulse oxymetry, non-invasive blood pressure systolic, FiO_2, temperature, Glasgow coma score total) are ranked amongst the top 80 percentile with the exception of **Glasgow coma score total**, which is ranked in the top 70 percentile.

Furthermore, to understand how the individual relationship types affect the predictive power of the GAT model. All the individual relationship types were evaluated on the

model. It was found that Admission - Drugs alone report an 85.8% AUROC, whereas Admission - Diagnosis achieved 81.7% AUROC, Admission - Vitals 71.3% AUROC and Admission - Labs 66% AUROC. This experiment of assessing individual relationship type was extended to understand the effect of a biased relationship in the structure by introducing a new relationship (**Demography**) by taking into account the patient similarity based on their demographic data **GAMER**(**G**ender, **A**ge-group, **M**arital status, **E**thnicity and **R**eligious status). It was found that the attention mechanism of the model corrected the effect of such a biased relationship.

8.2 Future Work

This research has investigated different aspects of representation learning on electronic health records using a graph neural network and provided new insights into the area of graph modelling, feature ablation, and helped to understand the effect of bias in the graph structure, evaluates the model's underlying predictors. However, there are still several areas for future research that can be built upon the findings of this study.

One potential area for future research is to extend the existing heterogeneous graphs, which presently involve the relationships between Admission and Labs/Drugs/Diagnosis /Vitals. One such extension would be connecting drugs with targeted proteins to find new therapeutic protein targets for a given drug or new drugs (for a given protein target) or connecting drugs based on known negative interactions to find adverse side effects. Similarly, diagnoses can be connected with symptoms, which can help identify patterns and connections that may not be immediately obvious. This can be used to identify new potential causes or risk factors for a condition or to find new ways to classify and diagnose a patient's condition. These relationships can be added using NLP from online sources of drug banks [1], and diagnosis-symptom data sources [2]. This study could contribute to a better understanding of such relationships and improve model performance on mortality prediction and their implications for embeddings generated by graph neural networks.

Another potential area for future research is to perform a time series analysis on the graph data and analyze how patients' mortality prediction is affected over different time periods. The change in AUROC, sensitivity and specificity over time could improve the reliability of predictions. The investigation could provide survived and expired patients' treatment trajectories which could be valuable in understanding how different treatment plans could affect mortality.

Another potential area is to integrate the reference ranges of labs results while training the GAT model and calculate the percentile ranking based on the edge weights. This would help to interpret the importance of variation of results in the prediction of mortality. For example, a reference range for creatinine levels in the blood, which indicate how well the kidneys function, typically ranges between 0.7-1.3 mg/dL for adult males and between 0.5-1.1 mg/dL for adult females [3].

[1] https://go.drugbank.com/

[2] https://accessmedicine.mhmedical.com/book.aspx?bookID=2715

[3] https://www.urmc.rochester.edu/encyclopedia/content.aspx?ContentTypeID=167&ContentID=creatinine_serum

Bibliography

[AAK+21] Fahad Shabbir Ahmad, Liaqat Ali, Hasan Ali Khattak, Tahir Hameed, Iram Wajahat, Seifedine Kadry, Syed Ahmad Chan Bukhari, et al. A hybrid machine learning framework to predict mortality in paralytic ileus patients using electronic health records (ehrs). *Journal of Ambient Intelligence and Humanized Computing*, 12(3):3283–3293, 2021.

[ABEDMB17] Aya Awad, Mohamed Bader-El-Den, James McNicholas, and Jim Briggs. Early hospital mortality prediction of intensive care unit patients using an ensemble learning approach. *International journal of medical informatics*, 108:185–195, 2017.

[Alh] Nawaf Alharbe. Impact of digitization of healthcare system in saudi arabia. *ICIC Express Letters*, 15(3):285–296.

[AY20] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

[BDLP20] Brent Biseda, Gaurav Desai, Haifeng Lin, and Anish Philip. Prediction of icd codes with clinical bert embeddings and text augmentation with label balancing using mimic-iii. *arXiv preprint arXiv:2008.10492*, 2020.

[BLM+06] Stefano Boccaletti, Vito Latora, Yamir Moreno, Martin Chavez, and D-U Hwang. Complex networks: Structure and dynamics. *Physics reports*, 424(4-5):175–308, 2006.

[BQE+21] Karan Bhanot, Miao Qi, John S Erickson, Isabelle Guyon, and Kristin P Bennett. The problem of fairness in synthetic healthcare data. *Entropy*, 23(9):1165, 2021.

[Cat11] Rick Cattell. Scalable sql and nosql data stores. *Acm Sigmod Record*, 39(4):12–27, 2011.

[CLL+20] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3438–3445, 2020.

[CMH+16] Jacob Calvert, Qingqing Mao, Jana L Hoffman, Melissa Jay, Thomas Desautels, Hamid Mohamadlou, Uli Chettipally, and Ritankar Das. Using electronic health record collected clinical variables to predict medical intensive care unit mortality. *Annals of medicine and surgery*, 11:52–57, 2016.

[Con13] Lynne M Connelly. Demographic data in research studies. *Medsurg Nursing*, 22(4):269–271, 2013.

[CPC+18] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):1–12, 2018.

[CXL+20] Edward Choi, Zhen Xu, Yujia Li, Michael Dusenberry, Gerardo Flores, Emily Xue, and Andrew Dai. Learning the graphical structure of electronic health records with graph convolutional transformer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 606–613, 2020.

[DCL18] Ali Davoudian, Liu Chen, and Mengchi Liu. A survey on nosql stores. *ACM Computing Surveys (CSUR)*, 51(2):1–43, 2018.

[EJCH05] Peter J Embi, Anil Jain, Jeffrey Clark, and C Martin Harris. Development of an electronic health record-based clinical trial alert system to enhance recruitment at the point of care. In *AMIA Annual Symposium Proceedings*, volume 2005, page 231. American Medical Informatics Association, 2005.

[EL14] Mehmet Ercan and Michael Lane. An evaluation of the suitability of nosql databases for distributed ehr systems. ACIS, 2014.

[ERESA+20] Nora El-Rashidy, Shaker El-Sappagh, Tamer Abuhmed, Samir Abdelrazek, and Hazem M. El-Bakry. Intensive care unit mortality prediction: An improved patient-specific stacking ensemble model. *IEEE Access*, 8:133541–133564, 2020. doi:10.1109/ACCESS.2020.3010556.

[FLNG+20] Neil M Ferguson, Daniel Laydon, Gemma Nedjati-Gilani, Natsuko Imai, Kylie Ainslie, Marc Baguelin, Sangeeta Bhatia, Adhiratha Boonyasiri, Zulma Cucunubá, Gina Cuomo-Dannenburg, et al. Impact of non-pharmaceutical interventions (npis) to reduce covid-19 mortality and healthcare demand. 2020.

[GAD+17] Thanos Gentimis, Ala' J. Alnaser, Alex Durante, Kyle Cook, and Robert Steele. Predicting hospital length of stay using neural networks on mimic iii data. In *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, pages 1194–1201, 2017. doi:10.1109/DASC-PICom-DataCom-CyberSciTec.2017.191.

[GBR21] Aya Gamal, Sherif Barakat, and Amira Rezk. Standardized electronic health record data modeling and persistence: A comparative review. *Journal of biomedical informatics*, 114:103670, 2021.

[GDDM14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[GL16] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[Ham20] William L Hamilton. Graph representation learning. *Synthesis Lectures on Artifical Intelligence and Machine Learning*, 14(3):1–159, 2020.

[HBM19] Steffen Hehner, Stefan Biesdorf, and Martin Möller. Digitizing healthcare–opportunities for germany. *Digital McKinsey. Online verfügbar unter: https://www. mckinsey. com/industries/healthcare-systems-and-services/our-insights/digitizing-healthcare-opportunities-for-germany, abgerufen am*, 9, 2019.

[HCW+18] Anahita Hosseini, Ting Chen, Wenjun Wu, Yizhou Sun, and Majid Sarrafzadeh. Heteromed: Heterogeneous information network for medical diagnosis. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 763–772, 2018.

[HLH+20] Nianzong Hou, Mingzhe Li, Lu He, Bing Xie, Lin Wang, Rumin Zhang, Yong Yu, Xiaodong Sun, Zhengsheng Pan, and Kai Wang. Predicting 30-days mortality for mimic-iii patients with sepsis-3: a machine learning approach using xgboost. *Journal of translational medicine*, 18(1):1–14, 2020.

[HYL17a] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[HYL17b] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017.

[JPS+16] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

[JWH+21] Dejun Jiang, Zhenxing Wu, Chang-Yu Hsieh, Guangyong Chen, Ben Liao, Zhe Wang, Chao Shen, Dongsheng Cao, Jian Wu, and Tingjun Hou. Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models. *Journal of cheminformatics*, 13(1):1–23, 2021.

[KDWZ85] William A Knaus, Elizabeth A Draper, Douglas P Wagner, and Jack E Zimmerman. Apache ii: a severity of disease classification system. *Critical care medicine*, 13(10):818–829, 1985.

[KKC+15] Megan E Keller, Sarah E Kelling, Douglas C Cornelius, Hafusat A Oni, and David R Bright. Enhancing practice efficiency and patient care by sharing electronic health records. *Perspectives in health information management*, 12(Fall), 2015.

[KW16] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[LC13] Xin Li and Hsinchun Chen. Recommendation as link prediction in bipartite graphs: A graph kernel-based machine learning approach. *Decision Support Systems*, 54(2):880–890, 2013.

[LCS+06] Pedro Larranaga, Borja Calvo, Roberto Santana, Concha Bielza, Josu Galdiano, Inaki Inza, José A Lozano, Rubén Armananzas, Guzmán Santafé, Aritz Pérez, et al. Machine learning in bioinformatics. *Briefings in bioinformatics*, 7(1):86–112, 2006.

[LGLS93] Jean-Roger Le Gall, Stanley Lemeshow, and Fabienne Saulnier. A new simplified acute physiology score (saps ii) based on a european/north american multicenter study. *Jama*, 270(24):2957–2963, 1993.

[LLP+20] Zheng Liu, Xiaohan Li, Hao Peng, Lifang He, and S Yu Philip. Heterogeneous similarity graph neural network on electronic health records. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1196–1205. IEEE, 2020.

[LM07] Andrew Lever and Iain Mackenzie. Sepsis: definition, epidemiology, and diagnosis. *Bmj*, 335(7625):879–883, 2007.

[LM17] Joon Lee and David M Maslove. Customization of a severity of illness score using local electronic medical record data. *Journal of intensive care medicine*, 32(1):38–47, 2017.

[LMD15] Joon Lee, David M Maslove, and Joel A Dubin. Personalized mortality prediction driven by electronic medical data and a patient similarity metric. *PloS one*, 10(5):e0127428, 2015.

[LTK+93] Stanley Lemeshow, Daniel Teres, Janelle Klar, Jill Spitz Avrunin, Stephen H Gehlbach, and John Rapoport. Mortality probability models (mpm ii) based on an international cohort of intensive care unit patients. *Jama*, 270(20):2478–2486, 1993.

[LXZ+21] Fuhai Li, Hui Xin, Jidong Zhang, Mingqiang Fu, Jingmin Zhou, and Zhexun Lian. Prediction model of in-hospital mortality in intensive care unit patients with heart failure: machine learning-based, retrospective analysis of the mimic-iii database. *BMJ open*, 11(7):e044779, 2021.

[LYH+20] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *European Conference on Computer Vision*, pages 541–556. Springer, 2020.

[MKB+20] Aurelie Mascio, Zeljko Kraljevic, Daniel Bean, Richard Dobson, Robert Stewart, Rebecca Bendayan, and Angus Roberts. Comparative analysis of text classification approaches in electronic health records. *arXiv preprint arXiv:2005.06624*, 2020.

[MMA+05] Rui P Moreno, Philipp GH Metnitz, Eduardo Almeida, Barbara Jordan, Peter Bauer, Ricardo Abizanda Campos, Gaetano Iapichino, David Edbrooke, Maurizia Capuzzo, and Jean-Roger Le Gall. Saps 3—from evaluation of the patient to evaluation of the intensive care unit. part 2: Development of a prognostic model for hospital mortality at icu admission. *Intensive care medicine*, 31(10):1345–1355, 2005.

[MQX14] Philip Moore, Tarik Qassem, and Fatos Xhafa. 'nosql' and electronic patient record systems: Opportunities and challenges. In *2014 Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pages 300–307, 2014. doi:10.1109/3PGCIC.2014.81.

[MWK+20] George Michalopoulos, Yuanxin Wang, Hussam Kaka, Helen Chen, and Alexander Wong. Umlsbert: Clinical domain knowledge augmentation of contextual embeddings using the unified medical language system metathesaurus. *arXiv preprint arXiv:2010.10391*, 2020.

[Nyk]

[PARS14] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[Pir07] Shariefuddin Pirzada. Applications of graph theory. In *PAMM: Proceedings in Applied Mathematics and Mechanics*, volume 7, pages 2070013–2070013. Wiley Online Library, 2007.

[PJA+21] Bart G Pijls, Shahab Jolani, Anique Atherley, Raissa T Derckx, Janna IR Dijkstra, Gregor HL Franssen, Stevie Hendriks, Anke Richters, Annemarie Venemans-Jellema, Saurabh Zalpuri, et al. Demographic risk factors for covid-19 infection, severity, icu admission and death: a meta-analysis of 59 studies. *BMJ open*, 11(1):e044640, 2021.

[PPC+15] Romain Pirracchio, Maya L Petersen, Marco Carone, Matthieu Resche Rigon, Sylvie Chevret, and Mark J van der Laan. Mortality prediction in intensive care units with the super icu learner algorithm (sicula): a population-based study. *The Lancet Respiratory Medicine*, 3(1):42–52, 2015.

[Pržo7] Nataša Pržulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2):e177–e183, 2007.

[PWH15] Adam Perer, Fei Wang, and Jianying Hu. Mining and exploring care pathways from electronic medical records with visual analytics. *Journal of biomedical informatics*, 56:369–378, 2015.

[Rif17] Nader Rifai. *Tietz textbook of clinical chemistry and molecular diagnostics*. Elsevier Health Sciences, 2017.

[RLRS+11] Vicent J Ribas, Jesús Caballero López, Adolf Ruiz-Sanmartín, Juan Carlos Ruiz-Rodríguez, Jordi Rello, Anna Wojdel, and Alfredo Vellido. Severe sepsis mortality prediction with relevance vector machines. In *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 100–103. IEEE, 2011.

[RRG18] David Reinsel-John Gantz-John Rydning, J Reinsel, and J Gantz. The digitization of the world from edge to core. *Framingham: International Data Corporation*, 16, 2018.

[RSN+14] Karthik Raghunathan, Andrew Shaw, Brian Nathanson, Til Stürmer, Alan Brookhart, Mihaela S Stefan, Soko Setoguchi, Chris Beadles, and Peter K Lindenauer. Association between the choice of iv crystalloid and in-hospital mortality among critically ill adults with sepsis. *Critical care medicine*, 42(7):1585–1591, 2014.

[RTV+21] Emma Rocheteau, Catherine Tong, Petar Veličković, Nicholas Lane, and Pietro Liò. Predicting patient outcomes with graph representation learning. *arXiv preprint arXiv:2101.03940*, 2021.

[RWE15] Ian Robinson, Jim Webber, and Emil Eifrem. *Graph databases: new opportunities for connected data*. " O'Reilly Media, Inc.", 2015.

[SBR18] Reza Sadeghi, Tanvi Banerjee, and William Romine. Early hospital mortality prediction using vital signals. *Smart Health*, 9:265–274, 2018.

[SDKGG20] Jens Schrodt, Aleksei Dudchenko, Petra Knaup-Gregori, and Matthias Ganzinger. Graph-representation of patient data: a systematic literature review. *Journal of medical systems*, 44(4):1–7, 2020.

[SDS+16] Mervyn Singer, Clifford S Deutschman, Christopher Warren Seymour, Manu Shankar-Hari, Djillali Annane, Michael Bauer, Rinaldo Bellomo, Gordon R Bernard, Jean-Daniel Chiche, Craig M Coopersmith, et al. The third international consensus definitions for sepsis and septic shock (sepsis-3). *Jama*, 315(8):801–810, 2016.

[SFED+12] Karim Sabbagh, Roman Friedrich, Bahjat El-Darwiche, Milind Singh, SANDEEP Ganediwalla, and RAUL Katz. Maximizing the impact of digitization. *The global information technology report*, 2012:121–133, 2012.

[SN20] Jessica AM Stothers and Andrew Nguyen. Can neo4j replace postgresql in healthcare? *AMIA Summits on Translational Science Proceedings*, 2020:646, 2020.

[TPV+16] R Andrew Taylor, Joseph R Pare, Arjun K Venkatesh, Hani Mowafi, Edward R Melnick, William Fleischman, and M Kennedy Hall. Prediction of in-hospital mortality in emergency department patients with sepsis: a local big data–driven, machine learning approach. *Academic emergency medicine*, 23(3):269–278, 2016.

[VCC+17] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *stat*, 1050:20, 2017.

[VFMV03] Alexei Vázquez, Alessandro Flammini, Amos Maritan, and Alessandro Vespignani. Modeling of protein interaction networks. *Complexus*, 1(1):38–44, 2003.

[VMT+96] J-L Vincent, Rui Moreno, Jukka Takala, Sheila Willatts, Arnaldo De Mendonça, Hajo Bruining, CK Reinhart, PeterM Suter, and Lambertius G Thijs. The sofa (sepsis-related organ failure assessment) score to describe organ dysfunction/failure, 1996.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[W+01] Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.

[WHA+21] Tingyi Wanyan, Hossein Honarvar, Ariful Azad, Ying Ding, and Benjamin S Glicksberg. Deep learning with heterogeneous graph embeddings for mortality prediction from electronic health records. *Data Intelligence*, 3(3):329–339, 2021.

[WJS+19] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032, 2019.

[WKN+18] Jason Walonoski, Mark Kramer, Joseph Nichols, Andre Quina, Chris Moesel, Dylan Hall, Carlton Duffett, Kudakwashe Dube, Thomas Gallagher, and Scott McLachlan. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. *Journal of the American Medical Informatics Association*, 25(3):230–238, 2018.

[WPC+20] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[YYHK22] Hai-Cheng Yi, Zhu-Hong You, De-Shuang Huang, and Chee Keong Kwoh. Graph representation learning in bioinformatics: trends, methods and applications. *Briefings in Bioinformatics*, 23(1):bbab340, 2022.

[ZCH+20] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.

[ZSH+19] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 793–803, 2019.

[ZTLT21] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2021.

[ZTXM19] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, 2019.