IMPROVING DATA ANALYSIS OF INCOMPLETE TABULAR DATA BY GRAPH REPRESENTATION FROM GRAPH NEURAL NETWORKS

PHAPHONTEE YAMCHOTE

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY (COMPUTER SCIENCE)
FACULTY OF GRADUATE STUDIES
MAHIDOL UNIVERSITY
2023

Thesis
entitled

IMPROVING DATA ANALYSIS OF INCOMPLETE TABULAR DATA BY GRAPH REPRESENTATION FROM GRAPH NEURAL NETWORKS

.....
Mr. Phaphontee Yamchote
Candidate

.....
Dr. Thanapon Noraset,
Ph.D. (Computer Science)
Major advisor

.....
Dr. Chainarong Amornbunchornvej,
Ph.D. (Computer Science)
Co-advisor

.....
Prof. .....
Dean
Faculty of Graduate Studies
Mahidol University

.....
Asst. Prof. Boonsit Yimwadsana,
Ph.D. (Electrical Engineering)
Program Director
Doctor of Philosophy Programme
in Computer Science
Faculty of Information and
Communication Technology
Mahidol University

Thesis
entitled

IMPROVING DATA ANALYSIS OF INCOMPLETE TABULAR DATA BY GRAPH REPRESENTATION FROM GRAPH NEURAL NETWORKS

was submitted to the Faculty of Graduate Studies, Mahidol University
for the degree of Doctor of Philosophy (Computer Science)
on
January 5, 2021

.....
Mr. Phaphontee Yamchote
Candidate

.....
.....,
Ph.D. (Computer Science)
Chair

.....
Dr. Thanapon Noraset,
Ph.D. (Computer Science)
Member

.....
.....,
Ph.D. (Computer Science)
Member

.....
Dr. Chainarong Amornbunchornvej,
Ph.D. (Computer Science)
Member

.....
Prof. .....
Dean
Faculty of Graduate Studies
Mahidol University

.....
.....,
Ph.D. (Computer Science)
Dean
Faculty of Information and
Communication Technology
Mahidol University

ACKNOWLEDGEMENTS

First

Phaphontee Yamchote

IMPROVING DATA ANALYSIS OF INCOMPLETE TABULAR DATA BY GRAPH REPRESENTATION FROM GRAPH NEURAL NETWORKS

PHAPHONTEE YAMCHOTE 6436198 ITCS/D

Ph.D. (COMPUTER SCIENCE)

THESIS ADVISORY COMMITTEE: THANAPON NORASET, Ph.D., CHAINARONG AMORNBUNCHORNVEJ, Ph.D.

ABSTRACT

Your abstract goes here.

IMPLICATION OF THE THESIS

Your thesis implication goes here.

KEY WORDS : MACHINE LEARNING / GRAPH NEURAL NETWORK

22 pages

CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

CHAPTER I
INTRODUCTION

1.1 Motivation

[Data Science and Tabular data]

In the contemporary landscape, data holds significant value in academia and business by data-driven paradigm. It drives the emergence of data science—an interdisciplinary field combining engineering, mathematics, statistics, and computer science. Applications of data science span various industries, uncovering concealed business opportunities. Research by Chatterjee et al. (2021) emphasizes its role in enhancing decision accuracy and fostering competitiveness. Among many kinds of data's format, tabular data is one of the formats that is widespread in domains like healthcare and manufacturing. It presents a structured means to store information.

[tell that missing data is a problem often met]

However, not only mining or modeling data but also data preparation is a task that practitioners in data science have to do. They spend much time in cleaning and organizing data. One of the common challenges is missing features which is the problem of absence for some values in the given datasets. This problem is relatively common in almost all research and all formats of data, especially in tabular data. It can significantly affect results obtained from the data [**?**] in sense of learning.

[causes of missing data]

There are many causes of missing data. First, the problem of measuring instruments is a cause that often occurs in a production line. Second, some missing data can emerge from the nature of data, such as different factors corrected between males and females in a clinical survey or the disappearance of a patient during the treatment process. Third, the lack of knowledge of the sample can also result in missing data. There may be the case of the sample giving a "don't know" answer

to an observer in some complicated question. Even independence of each measurement process (i.e. each feature) can be considered as a cause of missing data in sense of feature. That is, some feature can be determined by logical combination of two or more features. However, independent measurement make such information hidden.

[significance of missing data]

The missingness problem holds significant importance in machine learning and deep learning, primarily due to its computational implications. In these fields, including the cutting-edge domain of deep learning, algorithms often rely on fixed-size representation vectors, also known as embeddings. However, when dealing with missing data without proper preprocessing, a critical challenge arises as data cannot be directly fed into models due to the absence of vital information. From a theoretical perspective, machine learning involves the search for an optimal distribution within a hypothesis class, guided by a given training dataset. This search is essentially a quest to find the best model within an equivalent class of models. However, the presence of missing data complicates this task by weakening the inductive bias of learning, effectively expanding the search space for models. In simpler terms, a higher degree of missing data results in a larger equivalence class of models, making it more difficult to identify and learn the optimal models when working with datasets containing missing values.

[Brief Literature: existing methods which]

The most basic methods to handle this issue is imputation for the missing values. Many work also attempts to find methods to impute missing data in various way including distribution-based imputation and learning the missing values as well. To the best of our knowledge, there is no state-of-the-art imputation methods. Even in practice, the casewise deletion, i.e. removing the data points containing missing values, seems to be the most practical method if much more data is available.

However, Ma et al. (2018) [**?**] argue that imputation methods disregard data uncertainties, echoing the earlier notion of hidden information. Even

in some real-world case, missing values are their native structure which does not make sense to impute with value.

[Brief Literature: GNN potential]

From the review of literature, we found an interesting framework addressing this issue in different perspective by graph, called GRAPE [**?**]. They proposed a framework that can learn from data containing missing values without imputation for these missing subjects. GRAPE is a graph-based framework that table of data is represented by a graph which will be processed by an deep-network algorithm call graph neural networks (GNNs). This demonstrates the potential that representing data by graphs and using GNN algorithms can weaken the need of completeness of datasets. In other words, GNNs with graph data structure are more flexible in computation rather than using the fixed length of feature vector fed into algorithm directly.

Not only missing data, but GNNs are also used in many work that investigated approaches for modeling complete tabular data. There are both an idea of using one graph for the whole table like GRAPE does and idea of using a dataset of graphs which of each corresponds to each instance of table. It is a naive idea to use a big graph for the entire table to learn an hidden information inferred between row (instance) or column (feature). On the other hand, some work uses a graph of instance to capture hidden information inside the instance among its feature value.

why we use small graph:
computation and dependency

[Brief Literature: Gap between GNN and missing data]

GRAPE is an sample algorithm that allows avoidance of imputing. It is designed to capture the cooperating information between a node of instance and a node of feature that is observed for the instance and the captured information is aggregated to the node of instance to infer its corresponding label. It seems to utilize the flexibility of GNNs to omit the missing feature by not using the information from that feature node. However, some missing values may be informative in sense of structure. They should be utilized in prediction, not abandoned.

In contrast, the another idea to represent tabular data by graph, which represents each instance by a graph, is still not explored to address missing issue. So this motivates us to study how to address the missing data issue by the idea of graphs of samples. To the best of our knowledge, no work address the missing data by this idea. However, tabular data is not native graph data structure. So it comes up with the question of how to represent data by graphs and how to design GNN algorithm to process them.

[How we will fill the gap]

To fill the aforementioned gap, this work proposes the end-to-end GNN framework from tabular data to predicted label. The framework consists of two major modules: (1) graph construction and (2) prediction from graphs as shown in Figure 1.1.

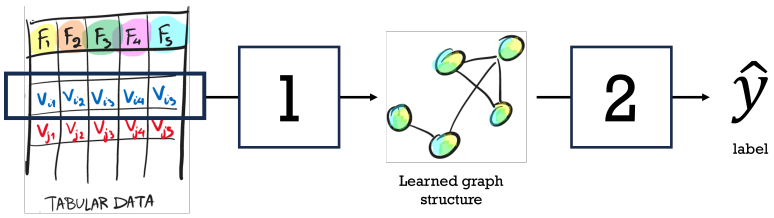

Figure 1.1: text

The graph construction module is for learning the structure of a graph of a given data point which is used as a graph representation of the instance. Since some missing values is caused by dependency of values of other features, e.g. not measurable attribute in male patient, we aim to develop a graph structure learning module so that it can keep dependency between features. Not only dependency, interaction between feature, i.e. the higher-order nonlinear operation between features is also an interesting characteristic of tabular data that can be considered as a missing value. Hence we also aim that this module of our work can also keep the interaction information in the learned graph structure.

On the other hand, we then develop the module making prediction from the learned graphs. This prediction is aimed to be robust for missing values, noise features, dependency and also interaction.

1.2 Research Objectives

Problem Statement

Missing data, which often occurs in many domains of tabular data, can lead to biased or inaccurate models if not handled properly. Current methods for handling missing data, such as imputation, can introduce additional assumptions and may not capture the underlying structure of the data. Moreover, doing so in the process of feature engineering takes much time and effort of practitioners. We seek to develop an approach that can effectively handle missing data in datasets without relying on process of data preparation for missing values such as imputation. Our model is also aimed to be able to learn the information of missing values in the sense of natural missing value

Objectives

The primary objective of this research is to investigate a GNN-based framework that can address the challenge of missing data. The further detail of our objectives is as follows:

1. develop an algorithm to construct spare graph structures for the representation of data that can be efficiently used for computation of the corresponding GNN prediction algorithm and maintain the interaction inside the data instance,
2. develop a GNN algorithm that can be trained by datasets containing missing values without any preprocessing, and also can predict the corresponding label of the given data that may contain missing values.
3. Validate the developed model both synthetic and real-world datasets.

1.3 Outlines

In Chapter 2, we explain more detail related to our work including graph representation, GNNs and related work. Readers can find deep detail and

also formal definition in the section preliminary of this chapter. And then, we describe how we perform this research in Chapter 3: methodology.

CHAPTER II
LITERATURE REVIEW

2.1 Preliminary

[Wrap-up 2 things and Missing data in tabular]

There are two major objects discussed in introduction chapter: (1) missing values and (2) GNNs. This work addresses the problem of missing values in tabular data by the use of GNNs.

In this section, we will give you more precise details of missing data and GNNs. Let us first introduce the formal notion of tabular data. We denote a table of data by $T \in \mathbb{R}^{m \times n}$ for the table of data of m samples and n features. When we refer to the ith instance, which is in the row ith, we use $T_{i:}$.

2.1.1 Missing Data

Missing data is the scenario that some feature values of a sample is absence by some reason such as quality of measuring equipment or dependency from value of other features. For example in clinical scenario, some features values of a patient may not be determined, even cannot, due to the gender of the observed patient. It can be classified into four categories [**?** **?**]: completely at random (MCAR), missing at random (MAR), missing not at random (MNAR), and mixed confounded missingness (MCM).

2.1.1.1 Missing Values Mechanisms

According to [**?**], Missingness can be categorized by dependency of missing features into 3 mechanisms: MCAR, MAR, MNAR, and MCM.

2.1.1.1.1 MCAR means that the missingness is random and has no relationship with the values of other variables in the dataset. For example, if we collect height data for a random sample of individuals, and some of them are absent from the data due to equipment malfunction, then the missingness is MCAR. In

this case, the missing data can be safely ignored in statistical analysis without introducing bias.

2.1.1.1.2 MAR means that the missingness is not random but can be explained by other variables in the dataset. For example, in a study of school performance, some students may not complete a survey on mental health due to embarrassment or discomfort, and their absence can be explained by their mental health status. In this case, the missing data are MAR, and ignoring it may lead to biased results. Therefore, imputation methods can be used to estimate the missing values based on the observed values and other relevant variables.

2.1.1.1.3 MNAR means that the missingness is not random and cannot be explained by other variables in the dataset. For example, in a study of employee salaries, some high-income earners may refuse to report their salaries, and their absence is related to the variable of interest but not to other variables in the dataset. In this case, ignoring the missing data may lead to biased results, and imputation methods may not be effective since the missing values are not predictable.

2.1.1.1.4 MCM means paragraph

Here is a formal definition of missingness mechanism provided in [**?**].

Definition 2.1. For precise describing, define the complete data matrix $X = (x_{ij})$ and the missingness indicator matrix $M = (m_{ij})$ where $m_{ij} = 1$ if x_{ij} is missing and $m_{ij} = 0$ if x_{ij} is observed. Assume that the rows (x_i) and (m_i) are independent and identically distributed over i. The missingness mechanism is characterized by the conditional distribution of m_i given x_i, say $f_{M|X}(m_i|x_i, \phi)$, where ϕ denotes unknown parameters. Let $x_{(1)i}$ denote the components of x_i that are missing for unit i.

1. **Missing Completely at Random** (MCAR) is the missing mechanism that missingness does not depend on the values of the data, missing or observed. That is, if for all i and any distinct values x_i, x_i^* in the sample space of X,

$$f_{M|X}(m_i|x_i, \phi) = f_{M|X}(m_i|x_i^*, \phi).$$

2. **Missing at Random** (MAR) is the missingness mechanism that missingness depends on x_i only through the observed components $x_{(0)i}$. Specifically, if for all i and any distinct values $(x_{(1)i}, x_{(1)i}^*)$ of the missing components in the sample space of $x_{(1)i}$,

$$f_{M|X}(m_i|x_{(0)i},x_{(1)i}, \phi) = f_{M|X}(m_i|x_{(0)i}, x_{(1)i}^*, \phi).$$

3. **Missing not at Random** (MNAR) is the missing mechanism that the distribution of m_i depends on the missing components of x_i.

And then, in 2023, Berrevoets et al. [?] proposed the new mechanisms .....

Definition 2.2 (Mixed Confounded Missingness)**.** content.

2.1.1.2 Patterns of Missing Data

There are many interesting patterns of missingness often met in real-world scenario. In this subsection, we explain these patterns of missing data and give some examples. And we still use the same notions from Definition 2.1.

1. **Univariate:** It is the case that missing data occurs only in a single variable. Specifically, let j^* be a fixed index of feature. We say the pattern of missing data is univariate if $j = j^*$ is a necessary condition of $M_{ij} = 1$. For example, paragraph

should specify trivial and nontrivial types
and should tell the reason why we handle each type

2. **Multivariate:** paragraph
3. **Monotone:**
4. **Disjoint:** Two subsets of variables never observed together.
5. **Latent:** A certain variable is never observed. Maybe it is even unobservable.

2.1.1.3 Traditional Methods to Handling Missing Values

There are many traditional methods to handle missing data varying from selection of only complete data to completion of incomplete values.

[Deletion]

The easiest way to handle this issue is keeping of only complete data. It seems to be the most practical way if we have sufficiently large enough complete data. It is obvious that we cannot use this method if the original data is incomplete because of its own nature

[Imputation: the most basic solution]

How will it be if the casewise deletion does not work because of limited size of data or the nature of data itself? One of the simple ways is imputation of these missing values. There are many methods of imputation varying from basic imputation by constant value or value of distribution [**?**] to model-based imputation by learning machine learning model [**?**].

However, these imputation may not reflect the actual dataset. [**?**] Actually, we do not even know the actual dataset. In addition, some missing behavior is also itself informative [**?**] such as unrating the product from a customer which may mean dissatisfaction but the customer avoids giving a reason directly. So imputation seems that we do not concern the hidden information about missing data. Moreover, Ma, et al. (2018) [**?**] commented that the imputation method

ignores uncertainties of data which means as the same as hidden information said earlier.

[More a bit Imputation but with learning from data]
Not only basic imputation methods, as aforementioned, but many work also investigated imputation based methods by learning from data. However, it still relies on the assumption that all missing values can be replaced as a number. But not all missing scenario can be imputed, for example, unrating the product. Also the case of some non measurable feature depending on others, the value 0 and a missing value are not equivalent. Moreover, these methods is independent from the learning process. In other words, the same model may be lead to different way by different imputation methods. The only way is to cross test all possible imputation methods to see whether which method is the best one for the desired model. This still requires much effort and time-consumption in the process of feature engineering.

2.1.1.4 Feature Interactions: the missing issue perspective

There is the another challenge in modeling tabular data which is not trivial related to missing issue. It is the scenario when two or more features have common influence to prediction, not only independent influence of each feature. For example, ... However, it can be considered as latent missing values as described in Subsection 2.1.1.2 because it is unobservable.

2.1.2 Graph Neural Network

As aforementioned in introduction chapter, to address missing data issue we want a framework that are flexible in validity or presence of feature values. In other words, how the model computes should be independent from the number of appearing values. Here, one of the efficient approaches that might fill this gap is to represent data by graphs and to use graph neural networks (GNNs).

Graph Neural Networks (GNNs) have emerged as a promising approach for modeling structured and semi-structured data, such as social networks [**?**], molecular structures [**?**], and knowledge graphs [**?**]. Traditional machine learning models often struggle to capture the complex dependencies in tabular data,

particularly when dealing with high-dimensional or incomplete datasets. GNNs, however, have demonstrated superior performance in various applications, as they effectively capture complex dependencies among nodes and edges in a graph [**?** , Kipf & Welling 2017].

2.1.2.1 Traditional Deep Learning

Before going deep further to detail of GNNs, let us introduce the broad definition of deep learning which is the fundamental idea of GNNs. After explaining these, we then move to the often-used layers for GNNs called message passing and graph pooling.

Besides traditional machine learning model like linear regression, k-nearest neighborhood, or decision tree, there is a model that is designed based on the computation of neural objects, called neuron, connecting together in the forms of network.

The neuron is a unit that mimics a biological neuron, consisting of an input (incoming signal), weights (synaptic weights) and activation function (neuron firing model). It was proposed due to the problem of nonlinear separable such as XOR problem. It can be defined formally as follows:

Definition 2.3 (Neuron)**.** A neuron is a quadruple $(\vec{x}, \vec{w}, \phi, y)$ where $\vec{x}^T = (x_0, \ldots, x_n)$ is the input vector, $\vec{w}^T = (w_0, \ldots, w_n)$ is the weights vector (in learning algorithm, they are called parameters), with $x_0 = -1$ and $w_0 = b$, the bias, and ϕ is an activation function that defines the outcome function $y = \phi(\vec{x}^T\vec{w})$.

Some neuron may be called with a specific name depending on the activation function ϕ inside itself. For example, if a neuron consists of the Heaviside function as activation, it is called a perceptron. And if it consists of logistic function, then it is said to be a sigmoid neuron.

Rather than models of a single neuron, some models is constructed from composition of many nodes whose outputs are fed into other layers of neurons. These models are called **neural networks** which is the base construction of deep learning models. It is defined formally as follows:

Definition 2.4 (Neural Networks)**.** Let us use the notations as follows:

$$\vec{x}^{(l)} = (x_1^{(l)}, \ldots, x_{d^l}^{(l)})^T$$
$$W^{(l)} = (w_{ij}^{(l)})_{i,j}$$
$$\vec{B}^{(l)} = (b_1^{(l)}, \ldots, b_{d^l}^{(l)})^T$$

and the convention that ϕ acts on each entries of its vector arguments as

$$\phi((x_1, \ldots, x_d)) := (\phi(x_1), \ldots, \phi(x_d))$$

A (feedforward) **neural network** model consisting of L hidden layers is an operation defined as

$$\vec{x}^{(l)} = \phi\left({W^{(l)}}^T \vec{x}^{(l-1)} - \vec{B}^{(l)}\right)$$

for $l = 1, 2, \ldots, L$.

Note that in this section, we describe only how they compute; i.e. the models. It is not how to learn their parameters which is an optimization problem or learning algorithm, including cost function, backpropagation, or gradient descent. For eager readers, these topics can be found in any textbooks about deep learning [**?**].

Deep learning is about how to design the architecture of construction from such neurons for each specific task from the input to the corresponding output. In other words, it is about concatenation of hidden layers of neurons. There are many well-known architecture in various domains such as recurrent neural networks (RNNs) [**?**] in sequential data structure like time series, convolution neural networks (CNNs) [**?**] for image processing, and transformers [**?**] for language.

2.1.2.2 Traditional Deep Learning towards GNNs

From the design of deep learning in the previous subsection, we see that deep learning models are designed in the perspective of grid data structure, i.e. Euclidean data structure. It relies on how we represent the inputs in the format of vectors or matrices. In image and text, they are still obvious to be a grid format. However, many data in real-world are not. For example, the relationship between people is data that is not trivial in how to represent by grid forms like vectors or matrices.

The example the we illustrated above is a sample for the format called graph data structure. What if we want to use a deep learning model for graph data structure?

2.1.2.3 Message Passing

content...

2.1.2.4 Graph Pooling

content...

2.2 Related Work

2.2.1 Graph Neural Networks for Tabular Data

GNNs have been applied to diverse types of tabular data, including healthcare [**?**] and financial data [**?**]. Their ability to model complex interactions and dependencies between variables [**?** **?**] and handle missing values [**?**] make them advantageous for tabular data analysis. However, designing GNNs for tabular data requires careful consideration of factors such as graph structure and feature engineering [**?**]. The graph structure should reflect the relationships between variables, while feature engineering should effectively transform input features into graph structures suitable for GNN models.

content...

2.2.2 Graph Neural Networks for Missing Data

content...

**********log**********

Missing data is significant because it can lead to biased model performance and inaccurate predictions. Handling missing data can be done through various methods such as statistical imputation, machine learning-based imputation, and deep learning-based imputation. Statistical methods such as mean imputation, regression imputation, and hot-deck imputation assume that the missing

values follow a certain statistical distribution, but they might not be accurate if the missing values are not random. Machine learning methods like k-nearest neighbor (KNN) imputation [**?**] and decision tree-based imputation [**?**] can be effective when the missing values have some patterns, but they might not work well if the missingness is too extensive. Deep learning methods such as autoencoders [**?**] and Generative Adversarial Networks (GANs) [**?**] can be used to impute missing values, but they require a large amount of training data and computational resources, and their performance might not be better than simpler imputation methods [**?**].

However, A limitation of imputation methods is that they assume that the missing values can be imputed based on the observed values and other relevant variables. However, in practice, we may not know the exact values of the missing data, and imputing them may introduce errors and bias in the analysis. Moreover, in some cases, the missing data itself may be informative, and imputing them may lead to loss of information. Therefore, it is important to carefully consider the nature of missing data and the suitability of different imputation methods before making any assumptions or decisions.

Recent research has shown that deep learning techniques can also be applied to handle missing data. One such approach is GAIN [**?**], proposed by J. Yoon et al. in 2018, which utilizes Generative Adversarial Nets (GANs) to generate plausible values for missing data based on the observed data. Another study by M. Smieja et al. [**?**] in 2019 also explored the use of neural networks for processing missing data, where a deep network was used to predict data containing missing values based on the observed values without imputation. In 2020, Ghorbani et al. [**?**] proposed a novel method called Embedding for Informative Missingness (EFIM), which uses a deep learning model to learn embeddings of the missing values that preserve their informativeness. The embeddings are then used to predict the corresponding label of input via the model that is learned together with the embedding. These recent studies suggest that deep learning-based methods can be effective for handling missing data, but they still ignore the dependencies of features on missing values.

In this work, we aim to develop a framework for handling missing data

that does not rely on explicit imputation. Specifically, we want to design a model that can predict the label of a given data point even if some of its features are missing, without necessarily imputing the missing values. If necessary, the model should also be able to reconstruct the missing parts of the input based on the other nonmissing features. Our proposed approach will leverage deep learning techniques to capture the complex relationships between the input features and the label, as well as the dependencies between the missing and nonmissing features. By doing so, we hope to provide a more accurate and robust solution to the missing data problem, while also preserving the integrity of the original data.

According to Grinsztajn (2022) [**?**], deep learning models have generally not performed as well as traditional machine learning models, particularly tree-based models, in tabular data analysis. As discussed in the previous section, there are several challenges that contribute to this underperformance. Recent advances in deep learning for tabular data have incorporated neural networks with architectures specifically designed for tabular data, TabNet [**?**] for example. Some work attempts to capture feature interactions [**?**] which is simply an operation among two or more features with respect to the output variable such as multiplication between two features, and handle missing data [**?**]. However, these methods can be computationally expensive. Table2Graph, for instance, uses a reinforcement learning approach that requires significant amounts of data and computation power to address feature interactions. GRAPE, on the other hand, models a table of data as a graph to handle missing data but does not address feature interactions.

Numerous studies have compared the performance of deep learning and traditional machine learning methods for tabular data. Some studies found that deep learning models outperform traditional machine learning models on certain datasets, such as those with high-dimensional and complex features, while others found that traditional machine learning models perform better on datasets with small to medium-sized features. For example, Ching et al. (2018) [**?**] found that deep learning models outperformed traditional machine learning models on a dataset with high-dimensional features. In contrast, other studies, such as Grinsztajn et al. (2022) [**?**], Olson et al. (2018) [**?**], and Fernández et al. (2014)

[**?**] found that traditional machine learning models, such as random forests and gradient boosting machines, outperformed deep learning models on certain tabular datasets. Despite these mixed results, it is clear that both deep learning and traditional machine learning methods have their strengths and weaknesses when it comes to tabular data analysis, and the choice of method should depend on the specific characteristics of the dataset and the research question at hand.

Graph representation learning has emerged as a promising approach for handling missing data. Notably, You et al. (2020) [**?**] proposed GRAPE, a framework utilizing GNNs for feature imputation and label prediction with missing data. GRAPE constructs a bipartite graph from the data matrix and formulates feature imputation as an edge-level prediction task and label prediction as a node-level prediction task.

Danel et al. (2020) [**?**] presented an approach for processing incomplete images as graphs, employing a spatial graph convolutional neural network (SGCN) to handle missing data without imputation. However, this method only considers spatial graph convolutions based on Euclidean distance, which may limit its ability to capture complex relationships between pixels. Alternative graph convolutions, such as spectral or attention-based, could offer greater flexibility and expressiveness.

GNN-based approaches for handling missing data have advantages over traditional imputation methods, as they do not rely on assumptions about missing values and can capture complex dependencies between features and missing data. Furthermore, the learned node or graph representations can be used for downstream analysis without imputing missing values, potentially leading to more accurate and interpretable results. Nevertheless, challenges remain in applying GNNs to handle missing data, including selecting appropriate graph construction methods, and GNN architectures, and evaluating their performance under various missing data scenarios.

CHAPTER III
METHODOLOGY

[In this section, you should remind your readers what the focus of your study is, especially the research aims. As we've discussed many times on the blog, your methodology needs to align with your research aims, objectives and research questions. Therefore, it's useful to frontload this component to remind the reader (and yourself!) what you're trying to achieve.]

some introduction some introduction

3.1 Problem Formulation

3.2 Research Methodology

To achieve our research objectives, we will use the following methodology:

3.2.1 Dataset

We will use benchmark datasets for regression and classification tasks that are commonly used in the literature. Specifically, we will use the datasets mentioned as benchmarks for tabular data in the work of Grinsztajn (2022) to ensure comparability with existing studies. We will also preprocess the data to handle missing values and normalize the features. Moreover, we also use synthetic datasets to be able to control the condition of interactions and missing values.

3.2.2 Graph Construction

Given a table of data $T \in \mathbb{R}^{n \times p}$ containing n instances (rows) and each instance has p attributes (columns). As depicted in Figure 3.1, the transformed graph of instance $i \in \{1, 2, \ldots, n\}$ is the graph $G_i = (V_{G_i}, E_{G_i})$ whose nodes $V_{G_i} = (F_1, \ldots, F_p)$ correspond to feature fields of the table. We call this graph a **feature graph**. The feature interactions are represented by edges.

In this work, defining E_{G_i} explicitly remains an open and under-explored question. As the ground-truth of feature interaction is typically unknown in real-world scenarios, we aim to create a model capable of learning feature interaction from the complete feature graph ($E_{G_i} = V_{G_i} \times V_{G_i}$) to enhance interpretability.

Figure 3.1: an example of construction of graphs from tabular data

3.2.3 Model Design

We will develop a novel GNN-based framework that takes datapoint-wise graphs as input. We will use PyTorch [**?**] and PytorchGeometric [**?**] to implement our proposed framework. The model will include attention mechanisms to identify important features and handle missing data. We will implement our proposed GNN framework and compare its performance with various baselines, including existing deep learning models and traditional machine learning models.

3.2.4 Performance Evaluation

We will use Mean Squared Error (MSE) as the evaluation metric for regression tasks and metrics from confusion matrices (e.g., accuracy, precision, recall, F1-score) as the evaluation metric for classification tasks. We will compare the performance of our proposed framework with existing deep learning-based models and traditional machine learning models.

We will conduct experiments to evaluate the performance of our proposed framework on benchmark datasets. Specifically, we will compare the performance of our proposed framework both without missing data and with missing data with the following models:

- Deep learning-based models: MLP (Multilayer Perceptron), CNN (Convolutional Neural Network), and DNN (Deep Neural Network).
- Traditional machine learning models: Random Forest, Gradient Boosting Machine, and Support Vector Machine. We will use five-fold cross-validation to ensure the reliability of the results.
- Graph-based deep models: GRAPE [**?**], Fi-GNN [**?**], Table2Graph [**?**]

3.3 Research Framework

REFERENCES

BIOGRAPHY

NAME	Mr. Phaphontee Yamchote
DATE OF BIRTH	26 August 1992
PLACE OF BIRTH	Bangkok, Thailand
INSTITUTIONS ATTENDED	Chulalongkorn University, 2012–2016
	Bachelor of Science (Mathematics)
	Chulalongkorn University, 2017–2021
	Master of Science (Mathematics)
	Mahidol University, 2021–20..
	Doctor of Philosophy (Computer Science)
E-MAIL	yamchote_p@outlook.com