Doctoral Dissertation

STUDY ON MATHEMATICAL FRAMEWORK FOR DATA ANALYSIS BASED ON MACHINE LEARNING

HIROKO NAGASHIMA

Graduate School of Science,
Tokyo Woman's Christian University

Doctoral Dissertation

STUDY ON MATHEMATICAL FRAMEWORK FOR DATA ANALYSIS BASED ON MACHINE LEARNING

機械学習に基づくデータ分析のための
数理基盤と方法論に関する研究

November 30, 2020

HIROKO NAGASHIMA

Graduate School of Science,
Tokyo Woman's Christian University

Abstract

In recent years, the volume and variety of data have increased, and the data from sensors and wearable devices are being used in a variety of fields, including a behavioral analysis of autonomous robots, customer trend analysis, and task management in factories. To improve the accuracy of the analysis, pre-processing (e.g., processing outliers and missing data, converting data format, combining multiple datasets, etc.) must be performed as needed. Pre-processing is a time-consuming task that requires more than 80% of the resources of a typical analytical process. Consequently, various methods of pre-processing have been proposed, such as manual methods using computer tools and automated methods using machine learning algorithms. However, these existing methods have two problems: i) they cannot infer the imputation data in all target types; ii) manual methods take time and effort for processing, and automated methods are difficult to customize for analysts, especially non-IT engineers.

To solve these problems, in this thesis, we propose a data mining framework called automated pre-processing for data mining (APREP-DM) and a data imputation method called automated pre-processing for sensor data (APREP-S). APREP-DM is characterized by a module to define business understanding schemes. We define the schemes before pre-processing. Therefore, a semi-automatic operation can be realized for the tasks. APREP-S is implemented into APREP-DM and performs automatic pre-processing. It selects the most optimal method among multiple pre-defined imputation methods, including statistics, time series analysis, and machine learning algorithms. To select the appropriate method, APREP-S ranks the candidates in the pre-defined methods using the probability model (i.e., function) defined in this thesis. The model parameters are learned during the training phase using the training data by maximizing the likelihood of the probability distribution. This process is conducted based on Bayesian inference and programming by example (PBE) approach. Therefore, the training and inference processes are iterated and conducted interactively. During the iteration, the human knowledge of the target environment can be incorporated into the model.

In this study, we verified the effectiveness and usefulness of APREP-DM by scenario-based and qualitative evaluations. This result shows that it is possible to perform

automatic pre-processing by clarifying business understanding schemes beforehand and that APREP-DM is a more well-balanced framework in sensor data analysis than other frameworks. For APREP-S, we conducted numerical experiments using two types of datasets: a climate dataset with long-term periodic data and a human activity dataset with short-term data. The metrics are the sum-of-squares and mean square errors. This result shows that APREP-S can select the appropriate imputation method according to the target features for both long- and short-term periodic data. In addition, the accuracy of the inference improved with each iteration of the training and inference processes. We conclude that the proposed data mining framework and data imputation method are efficient for data analysts and can reduce the resources required for the analytical process.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

In recent years, there has been an increase in the quantity and types of data available for analysis, including data acquired from sensors and wearable devices. Examples analysis applications using sensor data are customer trend analysis for shopping malls, autonomous behavior analysis for robots, and production management in smart factories. Because data analysis can support the knowledge of senior experts, it has drawn attention as a way to improve the productivity of non-experts in factories and to manage the utilization rate overall. Sensor data, in particular, tend to involve outliers and missing data rather than other types of data, such as structured data, because sensing systems generally use wireless networks and sensors having a battery, and process the data as time series. Therefore, it is necessary to check for outliers and missing data and to modify them as needed. These processes, termed pre-processing, use 80% of the resources of typical analytical processes, even for ordinary pre-processing methods [10].

An example of pre-processing is shown in Fig. 1.1. There are two types of data in the figure: weather data and person location data. The weather data has three columns: *time*, *id*, and *temperature*, which represent the measurement time, unique ID of each device, and location temperature at the measurement time, respectively. The time format is UNIX time and the sensing interval is 10 s. Outliers and missing data exist in this area. The person location data has four columns: *data*, *person_id*, x, and y, which represent the measurement time, unique ID of each person, horizontal axis, and vertical axis, respectively. The time format is year-month-day hour:minute, and the sensing interval is 10 min. These data formats differ according to device specifications, etc. Therefore, we need to transform one data format to another to integrate them into one dataset. Outliers and missing data are also required to be processed according to the aim of the analysis. In this case, the procedure for joining them is as follows:

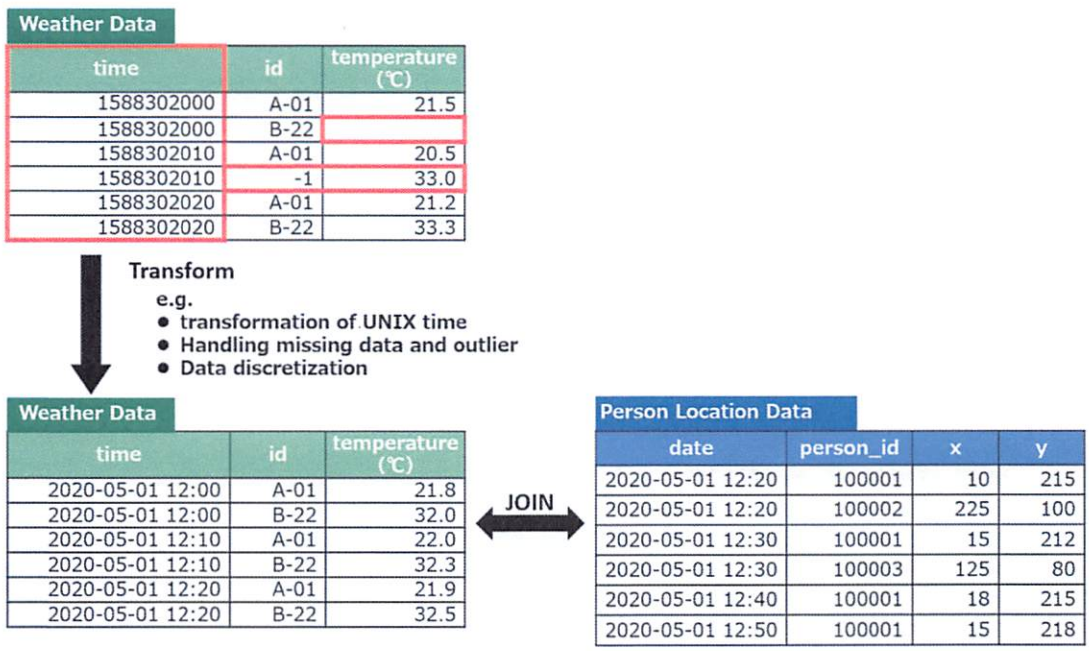

Fig 1.1. Example of pre-processing. Cells in red frames are transformed data.

- Transform UNIX time to year-month-day hour:minute on the weather data, e.g., 2020-05-01 12:20.
- Remove the row where $id = -1$ from weather data.
- Calculate the mean of the values in *temperature* column and input the value into the cell where *temperature* = NULL.
- Adjust the measurement interval of the weather data to that on the person location data and create a join key for integration.

After the pre-processing, it becomes possible to join the weather data with the person location data.

To perform such pre-processing, business understanding, which provides the goals, criteria, and knowledge of the analysis, plays a key role as well as IT skills. To demonstrate their importance, we specifically describe a typical data analysis process. First, all of the target data, including the business and sensor data, are obtained via computer networks and accumulated in a data store. Then, a data analyst conducts an iterable analysis process using extracted data from the store. An overview of the process is shown in Fig. 1.2. This process consists of five parts, which are considering the goal

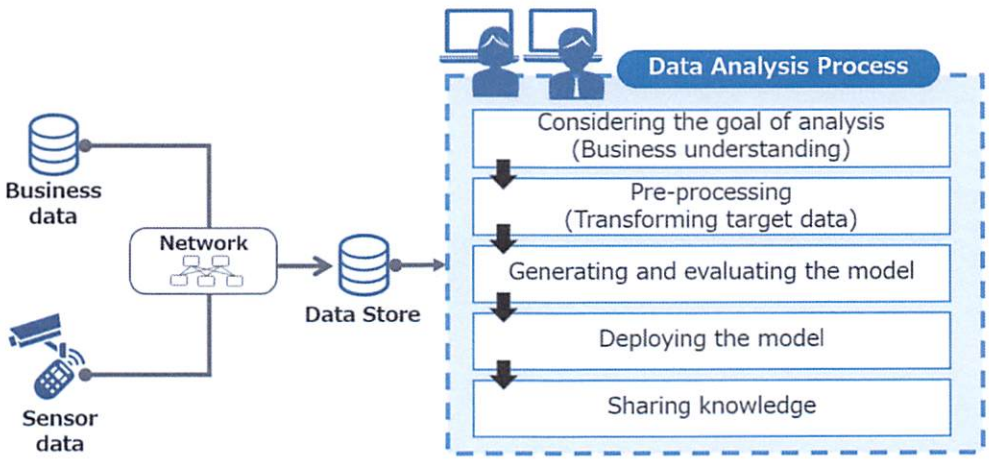

Fig 1.2. Overview of data analysis process.

of the analysis, pre-processing, generating and evaluating the analysis model (e.g., machine learning algorithms), deploying that, and sharing knowledge. Among them, the part of the considering the goal requires the analyst to understand the business, and the part of the generating and evaluating the method requires his/her IT skills, such as programming languages and machine learning algorithms. This means that the analyst is required to have both deep knowledge of the analysis (i.e., business understanding) and IT skills. In reality, there are not enough such analysts because data analysis tasks are currently increasing in all business fields.

1.2 Motivation

As mentioned in the previous section, to improve the accuracy of data analysis, pre-processing must be performed as needed. Here, pre-processing is a time-consuming task; therefore, various pre-processing methods have been proposed, such as manual methods using computer tools and automated methods using machine learning algorithms. However, these existing methods have two problems: 1) they cannot infer the imputation data in all target types; 2) manual methods take time and effort, and automated methods are difficult to customize for analysts, especially non-IT engineers. Solving these problems is our motivation for this study.

In this thesis, we focus on the pre-processing scheme from two aspects, namely, automation and integration of human knowledge with automated methods. Automation reduces the workload of the analytical process and allows analysts to perform the

Table 1.1. Comparison of manual, automated, and hybrid methods.

	Manual method	Automated method	Hybrid (Our proposal)
Customization	Easy	Difficult	**Easy**
Automation	Nothing	All	**Almost**
Accuracy	High (*)	Normal	**High**

(*) What we can work manually is limited.

process if they do not have sufficient IT skills. However, there are pre-processing tasks that cannot be performed automatically because of a lack of business understanding and trial-and-error processes. Moreover, it has been reported that the accuracy of automated methods (e.g., machine learning) is too low without pre-processing [11]. Here, we consider such a machine learning model as an automated method. When using the model, we must generate and update it according to the target features and business processes. At that time, a complete automated method cannot involve trial-and-error processes in the model; therefore, the analyst needs to perform pre-processing manually and then generate the model. In general, the analyst writes programming codes and uses transformation tools. Nevertheless, this is not enough to reduce the workload. Writing macros/scripts and operating computer tools are too difficult for analysts who are not familiar with programming [12] [13].

From these viewpoints, we adopt a hybrid method of manual and automation in this study. Table 1.1 shows the comparison results of these methods. Manual methods, such as using data transformation tools, are easier to customize for pre-processing and have higher imputation accuracy, but they have a greater workload. In addition, they are difficult to use for non-IT engineers. In contrast, automated methods, such as those using machine learning algorithms, do not require manual operations after generating the model, but generating an accurate model is difficult if we cannot collect data with sufficient quality and quantity. Our target is a hybrid method that combines the advantage of manual and automated methods. The manual part can realize customization easily and the automated part can reduce workload.

1.3 Objective

To realize the hybrid method described in the previous section, in this thesis, we propose a data mining framework including business understanding schemes and a semi-automated data imputation method for the framework. The objectives of this thesis are as follows:

- Proposing the framework and the imputation method that supports non-IT engineers to perform data analysis tasks.

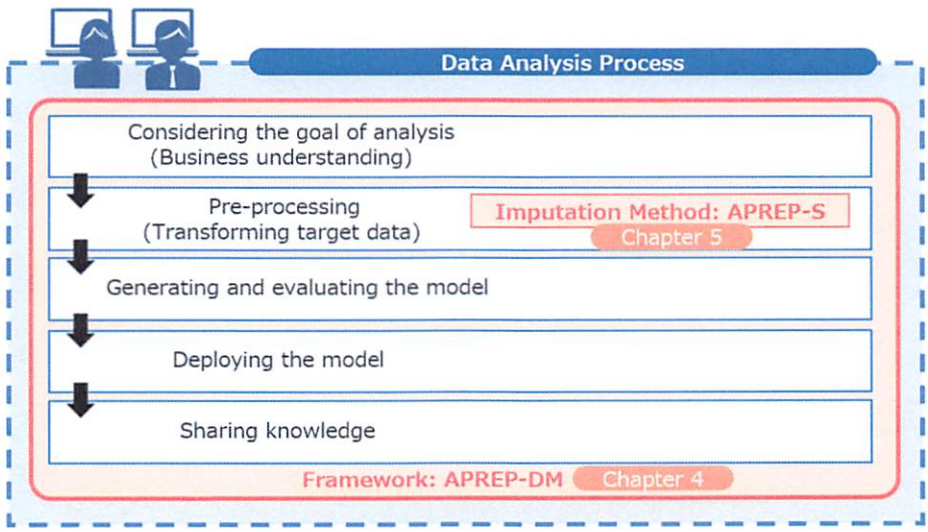

Fig 1.3. Overview of this study based on Fig. 1.2.

- Introducing the method generating and updating the model for data imputation based on Bayesian inference and programming by example approaches.
- Verifying the effectiveness and usefulness of the proposed framework by comparing it with the well-known framework and imputation methods.

1.4 Overview

An overview of this study is presented in Fig. 1.3. In this section, we describe the outline of the proposed framework, the imputation method, and its application.

1.4.1 Data Mining Framework

For the data mining framework, we propose the framework called automated pre-processing for data mining (APREP-DM) based on cross-industry standard process for data mining (CRISP-DM) [3], which is a well-known frameworks. Followings are the reasons why pre-processing is a time-consuming task: 1) data are of various types and formats; 2) data analysis methods are diversified; and 3) numerous pre-processing requirements exist [2]. Therefore, to reduce the pre-processing tasks for the framework, we define processes that can be executed automatically.

Furthermore, we verify the effectiveness and usefulness of APREP-DM through scenario-based and qualitative evaluations.

1.4.2 Data Imputation Method

For the data imputation method, we propose a hybrid method of automatic and manual processes that handles outliers and missing data, called automated pre-processing for sensor data (APREP-S). This method selects the most optimal method among multiple pre-defined imputation methods, such as statistics, time series analysis, and machine learning algorithms. To select the appropriate method, APREP-S ranks the candidates among the pre-defined methods using the probability model (i.e., function) defined in this thesis. The model parameters are learned during the training phase using the training data by maximizing the likelihood of the probability distribution. This process is performed based on Bayesian inference and programming by example (PBE) approach. Therefore, the training and inference processes are iterated and conducted interactively. During the iteration, the human knowledge of the target environment can be incorporated into the model.

For APREP-S, we conducted numerical experiments by using two types of datasets: a climate dataset with long-term periodic data and a human activity dataset with short-term data. The metrics used are the sum-of-square and mean square errors.

1.4.3 Application

Here we present a brief description of the use case of the proposed framework in the context of an actual business scenario, where a machine learning model is used for data analysis. In a typical business environment, IT engineers work for the IT department, business experts (in this thesis, we term them as project experts) work for the marketing and manufacturing department, etc. Therefore, they usually work in different sectors, sites, and fields. However, the machine learning model is developed in the IT department by the IT engineers, and it is used on each project site by the project experts. In such a scenario, it is necessary to understand the operation, adaptation, and updating of the model for each site [14]. The conditions, environments, and features of different project sites, such as climate, floor mapping, and project rules, vary. Although it would be desirable to assign IT engineers to every project and site, it is impractical, owing to the limited number of IT engineers. Furthermore, it is aimed to reduce costs and human resources [15] [16].

To this end, currently, there are cyber-physical systems such as Industry 4.0 [17] and digital transformation systems [18]. In such systems, project experts can use the machine learning models on-site because the IT engineers can generate suitable models

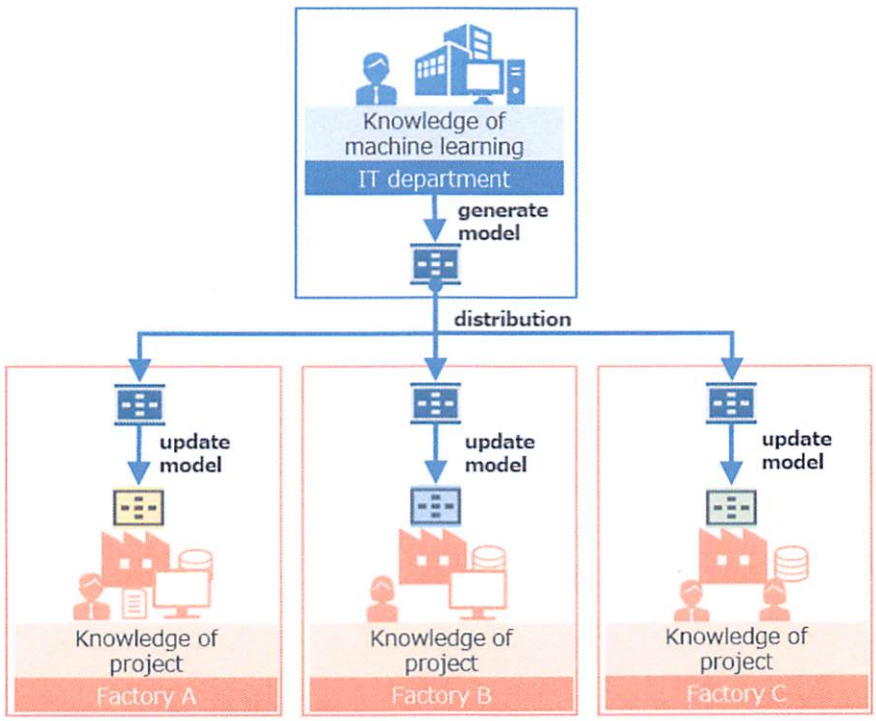

Fig 1.4. Image of distribution of machine learning model.

and deliver them via computer networks to the target sites. An image of an example scenario (a factory case) is illustrated in Fig. 1.4. The IT engineer first generates an original model, and the project expert updates it according to the features of the site. The proposed framework can be applied to the operating and updating processes. A specific use case of digital manufacturing is described in Appendix A.

1.5 Structure of the Thesis

This thesis is organized as follows:

Chapter 2 reviews the research related to this study. We select four well-known data mining frameworks and depict their software architectures as diagrams. Among them, CRISP-DM is the basis of the proposed framework, APREP-DM. Furthermore, we classify the data imputation methods into three categories: deterministic rule-based methods, time-series analysis, and machine learning-based methods. These methods

can function as candidates for use in APREP-S.

Chapter 3 describes the main learning algorithms used in our probability model, APREP-S. This chapter discusses the following topics: multi-class Bayesian logistic regression, hidden Markov model, and k-Shape. Additionaly, we introduce programming by example, which constitutes the concept of learning in APREP-S.

Chapter 4 develops the data-mining framework, APREP-DM. The development comprises six steps in its workflow: business understanding, data understanding, preprocessing, modeling, evaluation, and deployment. The design results of APREP-DM are presented in this chapter. Futhermore, we present our contributions to data analysis tasks by conducting scenario-based and qualitative evaluations.

Chapter 5 forms the theoretical core of this thesis and presents the proposed imputation method, APREP-S. We discuss the proposed probability model for the selection of the optimal method among the candidates, which is based on Bayesian inference and programming by example approach. This chapter also presents the experimental results using two types of datasets and reveals the effectiveness of the proposed scheme.

Chapter 6 concludes the thesis and outlines our future works.

Chapter 2

Related Work

There are two main research areas in this study: 1) the data mining framework and 2) the data imputation method. Related work for each is described in the following.

2.1 Data Mining Framework

We focus on four well-known frameworks for data mining. The characterizations of these frameworks are summarized in Table 2.1. First, knowledge discovery in the database (KDD) [1] [19] is the oldest data mining framework proposed by Fayyad et al. in 1996. This framework can be repeated between any steps, while the analyst needs to consider whether or not to return. Next, CRISP-DM is a cross-industry standard process for data mining [3] and is the name of the consortium name for data mining. This framework can clarify the project's aim and criteria at first, but it does not implement steps for handling outliers. Last, SEMMA is taken from the initials of sample, explore, modify, model, and assess [4], and ASUM-DM stands for analytics solutions unified method for data mining and predictive analytics [5] [20]. These frameworks are for data-mining products. SEMMA has a step for sampling data at first so that a trial-and-error approach can be used, but it does not have any analysis steps, such as related to business understanding and sharing knowledge. ASUM-DM is a simple iteration by an integration step from business understanding to model evaluation, but, like CRISP-DM, there are no steps for treating outliers.

We describe each of these specific frameworks in the following.

2.1.1 KDD

KDD involves nine steps in one cycle. A notable feature is that a step in the process may be repeated if necessary. The framework is illustrated in Fig. 2.1 and comprises

Table 2.1. Characterization of well-known frameworks.

	Feature	Problem
KDD	iteration between all steps if necessary	complex flow, as all steps can iterate to every step
CRISP-DM	clarifying the priority and the criteria of the project	effect of outliers
SEMMA	trial-and-error approach by sampling	business understanding and sharing knowledge does not exist
ASUM-DM	easy iteration by integration model generation step for big-data analysis	effect of outliers

the following steps:

1. Learning the application domain: understanding the application domain and the business aim of the analysis.
2. Creating a target dataset: extracting data or sampling to generate a target dataset for analysis.
3. Data cleaning and pre-processing: removing noise, mapping missing data, or transforming time-sequence information.
4. Data reduction and projection: identifying data trends by dimensionality reduction or using transformation values, i.e., data reduction or data projection.
5. Choosing the data mining function: selecting the model to achieve the final goal of the analysis among data integration, classification, or clustering.
6. Choosing the data mining algorithm: evaluating the model and considering the analysis model.
7. Data mining: running the model, e.g., using regression or clustering.
8. Interpreting the results: understanding the results and visualizing patterns and the model.
9. Using the acquired knowledge: documenting the outcome and checking for conflicts with earlier results.

Within the KDD framework, steps may be repeated and improved through successive iterations. However, this can increase the complexity significantly owing to the need to consider the return points at each step. Therefore, the KDD framework can be very time consuming.

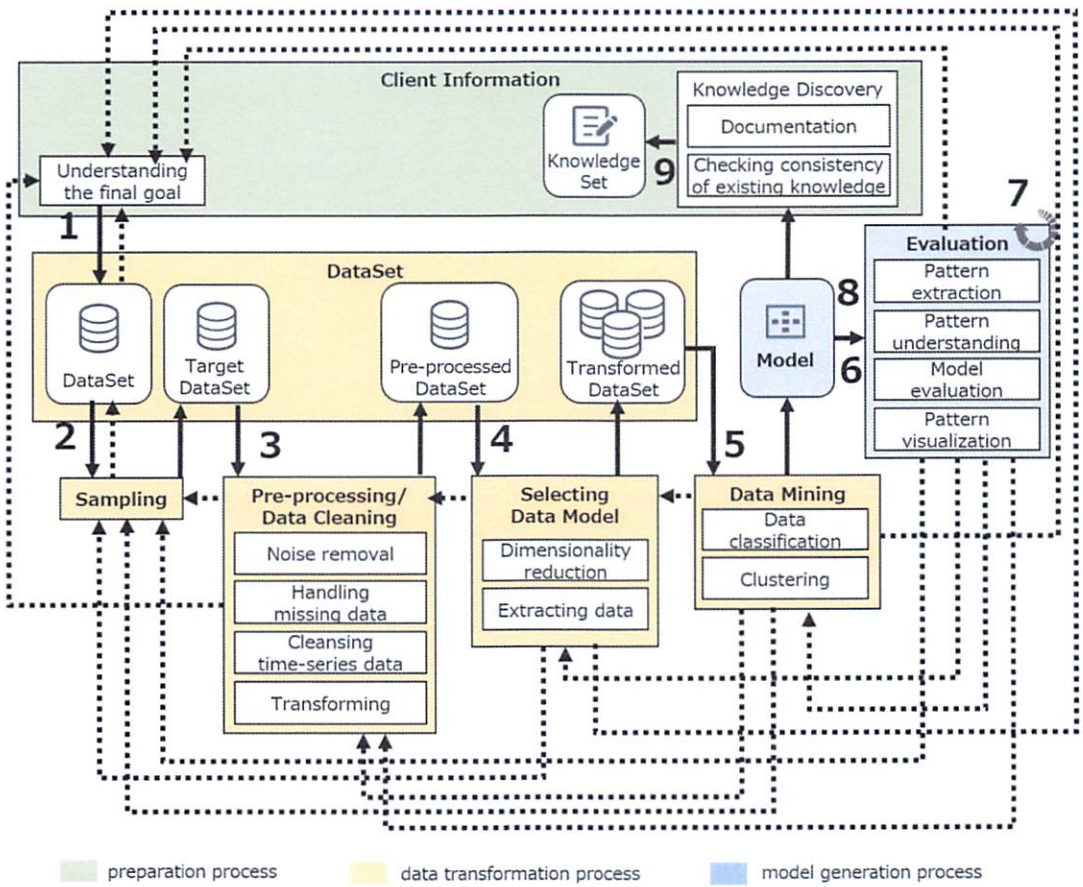

Fig 2.1. Overview of KDD framework (drawing based on literature [1], Figure 1). ©2019 IEEE in literature [2]

2.1.2 CRISP-DM

CRISP-DM was proposed by the CRISP-DM consortium of companies (including NCR, SPSS, and DaimlerChrysler, and others) that perform data mining. There are six steps in each cycle. Its features are 1) an initial clarification of the priority and end-goal criteria and 2) the inclusion of iterations between business understanding and data understanding and between data preparation and modeling The framework is illustrated in Fig. 2.2 and comprises the following steps:

1. Business understanding: clarifying the client's aim and defining the priority and success criteria.
2. Data understanding: understanding the data used within the project and evalu-

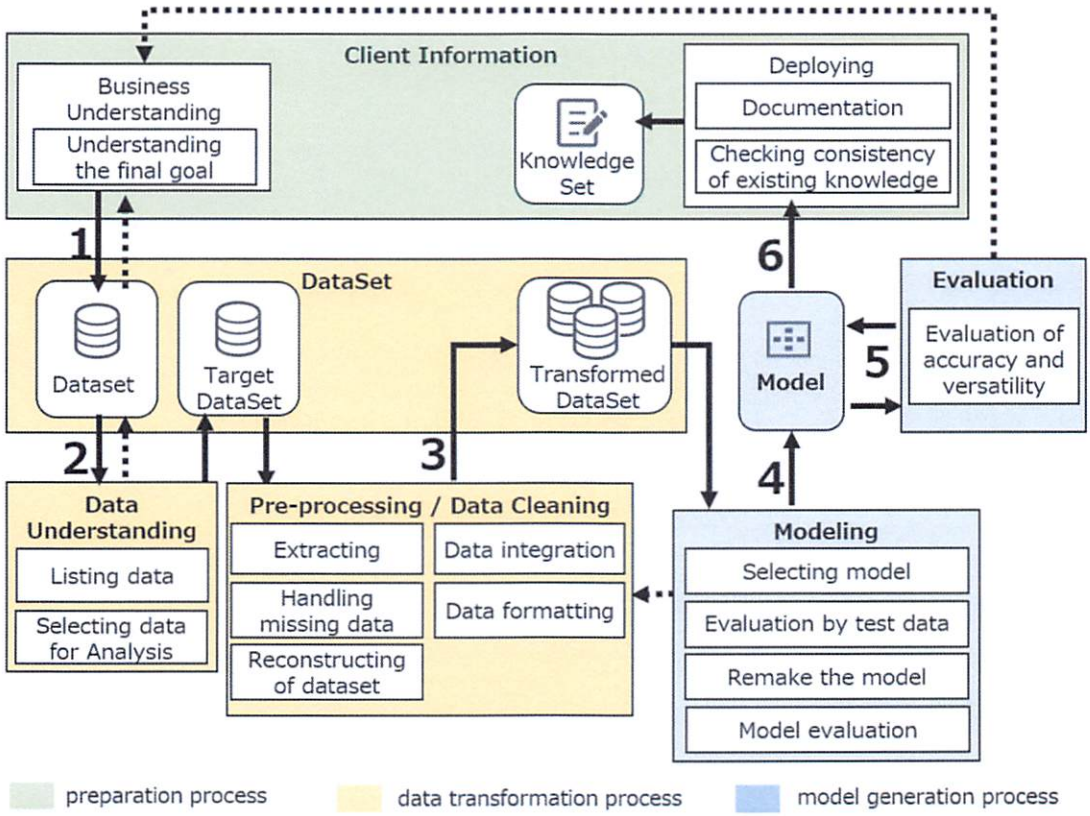

Fig 2.2. Overview of CRISP-DM framework. (drawing based on literature [3]. ©2019 IEEE in literature [2]

ating the data if required.

3. Data preparation: performing the necessary data transformations, e.g., extracting target data, handling missing data, and reconstruction of the dataset.
4. Modeling: selecting the model, e.g., decision tree or neural network.
5. Evaluation: using an application to evaluate model accuracy and versatility.
6. Deploying: summarizing the process and sharing knowledge.

CRISP-DM involves a step in which the client's priority and the success criteria of the project based on business understanding are decided. However, this framework does not treat outliers in the data-preparation step. Consequently, the potential impact of outliers on the project outcome makes CRISP-DM unsuitable for sensor-data analyses. Although the CRISP-DM framework does not treat outliers, the CRISP-DM consortium considers outliers and missing data.

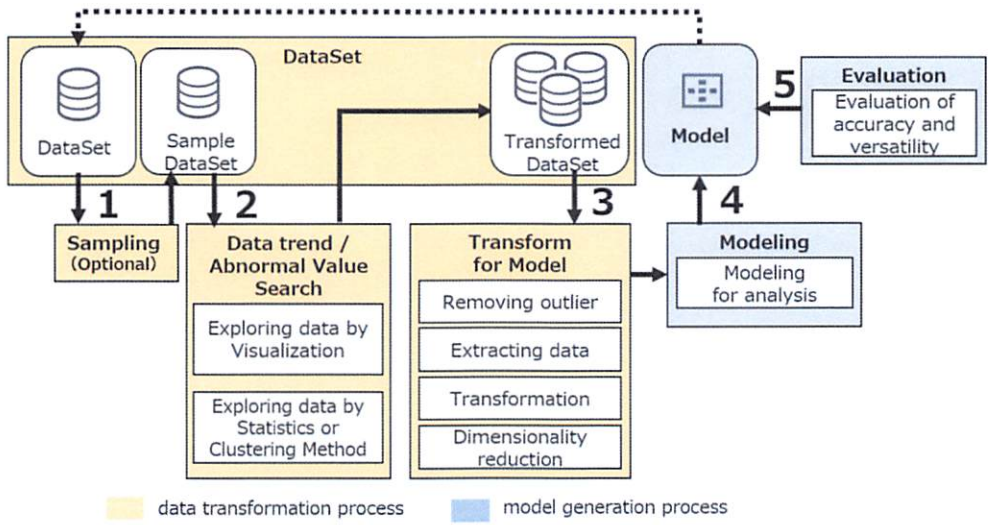

Fig 2.3. Overview of SEMMA framework. (drawing based on literature [4].) ©2019 IEEE in literature [2]

2.1.3 SEMMA

SEMMA, a framework proposed by the SAS Institute, involves five steps in one cycle. This framework was designed for a data-mining product of the SAS Institute called SAS Enterprise Miner. Therefore, the steps are classified according to the product's functions. The features of this framework are 1) data extraction for sampling and 2) exploration of data trends. SEMMA is often used by enterprise situations. The framework is illustrated in Fig. 2.3 and comprises the following steps:

1. Sampling: extracting a portion of a large data set by random sampling (an optional step).
2. Exploration: understanding data trends by visualization, statistics, clustering, etc.
3. Modification: adding new items, extracting or transforming data; if necessary, removing outliers or reducing dimensionality.
4. Modeling: making a model for analysis methods, e.g., decision trees or neural networks.
5. Assessment: evaluating the model in terms of usability, reliability, or accuracy.

SEMMA includes a data-sampling step when handling a large dataset and is therefore amenable to a trial-and-error approach to data mining. However, it does not

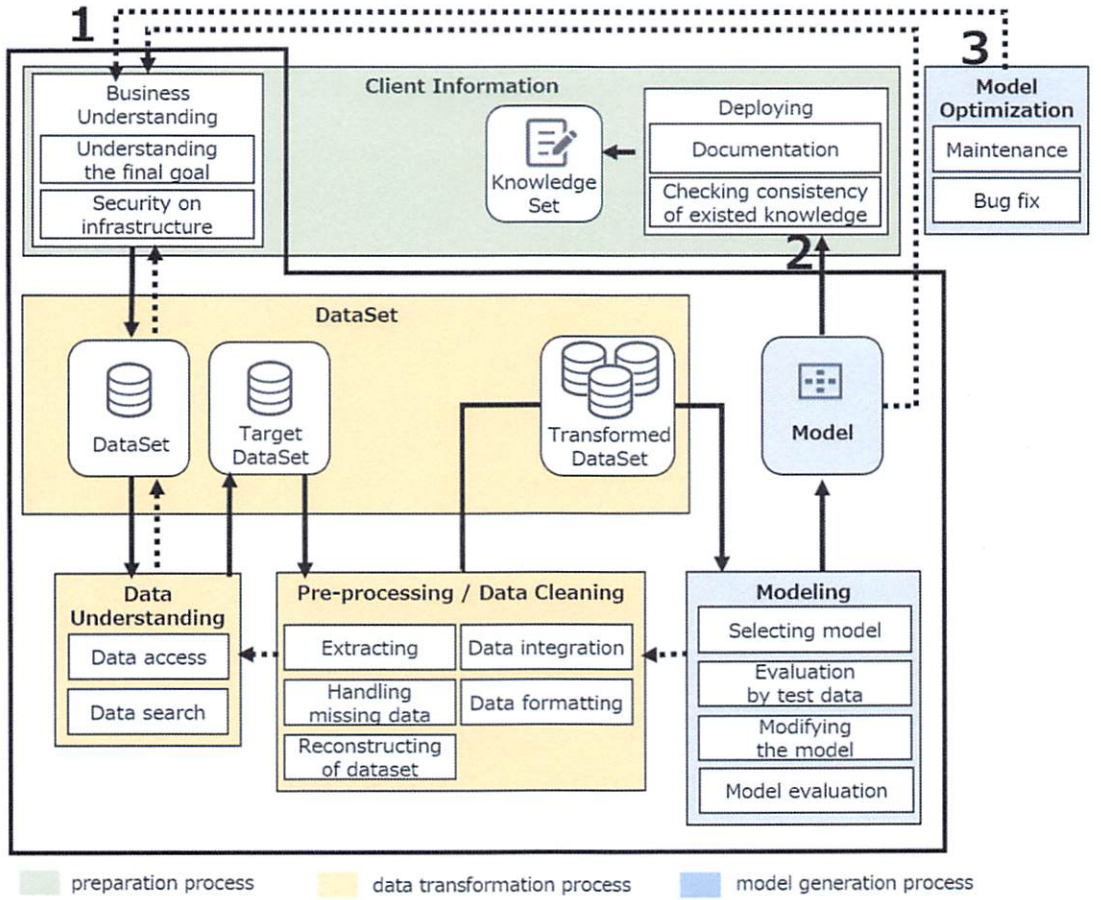

Fig 2.4. Overview of ASUM-DM. (drawing based on literature [5].)

include the business understanding and deploying steps. Client information and knowledge must therefore be managed outside the framework.

2.1.4 ASUM-DM

ASUM-DM is a framework proposed by IBM based on CRISP-DM. It is designed for SPSS, which is a data modeling product. This model improves on the maintenance process using accumulating knowledge by adding deployment to the iteration cycle. There are five steps in ASUM-DM. The features of this framework are 1) integration of three steps—analyze, design, and configure & build, because to iterate these three steps is natural in the data mining process, 2) considering security, and 3) describing maintenance after model generation and bug fixes.

Table 2.2. Types of missing data.

Type of missing data	Feature of occurrence
MCAR	completely random
MAR	not completely randomly
MNAR	depends on the missing data themselves

The framework is illustrated in Fig. 2.4 and comprises the following steps:

1. Analyze, Design, and Configure & Build: understanding the final goal and requirements of the customer (business understanding) and building a security-conscious infrastructure and understanding the data to avoid unexpected errors and iterating the model generation and evaluation.
2. Deploy: accumulating knowledge and continuing the analysis operations by analysts and deploying the knowledge gained on a global level.
3. Operate & Optimize: maintaining the correct state of the process, such as model maintenance and bug fixes.

The inclusion of the deploy step in the iterations of the model is the point of ASUM-DM. This is the knowledge of the model generation and the accumulation of the analysis. In practice, it is common for a system to continue operating for several years once it has been created. Therefore, the accumulation of knowledge is an important factor. However, because ASUM-DM is based on CRISP-DM, there is no description of the outliers, but the effects of the outliers may be incorporated into it.

2.2 Data Imputation Method

Missing data are often categorized into the following three types [21] [22]: missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR). Their features are listed in Table 2.2. MCAR is the case in which missing values occur completely randomly, and their occurrences do not depend on the observable data or the missing data. In other words, this is the case in which the relevant data items to be used in the analysis are complete or the data pattern only includes items that can be estimated from other data even if the items themselves are missing. MAR is the case in which the occurrence of missing values is not completely random, and the occurrence of missing data depends on the observed data. For example, if an employee does not self-report his or her age, the value is a missing value. However, age does not actually depend on the occurrence of the missing value. MNAR is a case in which the occurrence of missing values depends on the missing data themselves. For

Table 2.3. Classification of imputation method.

	Feature	Problem
Calculated based on predetermined equation	easy processing by decided rule	difficult to insert human knowledge
Inference by time-series analysis	able to analyze periodic data	difficult to insert human knowledge
Inference by machine learning	better accuracy by iteration of model training	difficult to insert human knowledge

example, suppose that, in a survey, an expert answers that he or she has an experience, while, however, a nonexpert does not answer that question. In this case, the missing data depend on whether or not the responder is an expert. As the missing data occur completely randomly in a sensor network, we consider the MCAR data in this study.

We classify related works on data imputation methods into three categories: 1) calculations based on predetermined equations, 2) inference by time-series analysis, and 3) inference by machine learning. We summarize these methods in Table 2.3. The method of using a rule means calculating by using the decided equation beforehand, for example, the mean of the entire data in the column, the median of the entire data in the column, or the input of a fixed value zero. It is easy to develop a program because we can calculate by using the same rule. The method of inferring by time series analysis is suitable to calculate periodic data. It can output the data which includes time-series trend. The method of inference by time-series analysis is suitable for calculating periodic data. It can output data that include a time-series trend. The method of inference by machine learning becomes more accurate with further iterations of model training. It can include the features of other data. However, it is difficult to integrate human knowledge into each of these methods because there is no room for customization. Therefore, we propose a method to select an optimal method from among these data imputation methods.

We describe specifically these categories in the following.

2.2.1 Calculated based on Predetermined Equation

In the data imputation method using rules, we can use a tool for handling missing data. First, the methods are described, and then the tools.

Single Imputation

The single imputation method imputes the unique values that are calculated according to a specific predefined rule from the collected data. As mentioned above, this could

be the mean of the entire data in the column, the median of the entire data in the column, or inputting a fixed value of zero. One of the most popular methods of single imputation is spline interpolation [23]. Spline interpolation is a method for drawing smooth curves through equally spaced data. It fits a polynomial to the specified data points and obtains a curve that passes through all specified points. No data point is lost because changing the coefficients of the polynomial changes the curve without moving any of the data points. Linear spline interpolation is a polygonal curve because the function between connecting data points is assumed to be a linear function. A quadratic or higher spline interpolation results in a differentiable curve. For example, a cubic polynomial is expressed by the following equation:

$$S(x) = \begin{cases} s_1(x) & \text{if} \quad x_1 \leq x < x_2 \\ s_2(x) & \text{if} \quad x_2 \leq x < x_3 \\ & \vdots \\ s_{n-1}(x) & \text{if} \quad x_{n-1} \leq x < x_n \end{cases} \tag{2.1}$$

$$s_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \tag{2.2}$$

where $i = 1, 2, ..., n - 1$.

A single imputation method can calculate all imputation values automatically provided a rule is defined beforehand. However, it cannot change a single value in the list of imputed values. Therefore, it lacks the scope of customization.

Data Preparation Tool

There are data preparation tools in which the analyst defines and develops the pre-processing processes on his or her own and checks the data profile to determine whether there are missing data, outliers, or the presence of inconsistencies in the format or spelling. Analysts can reduce the number of tasks that must be performed by using tools such as OpenRefine [24] and Trifacta Wrangler [25]. These can assist analysts in sorting, aggregating, and detecting data that need to be transformed according to the GUI. Moreover, analysts record the processing log. Therefore, the process can be automatically rerun if the process flow is the same.

However, analysts need to perform maintenance on their own when the flow changes. In addition, they must consider the configuration of the imputed values while they can remove data easily. Therefore, the available tools are not suited for pre-processing of imputed data.

2.2.2 Inference of Time Series Analysis

One of the most well-known methods of time-series analysis is to use the generalized additive model (GAM). The GAM generates a nonlinear function by adding multiple functions together [26]. The ith observation of the GAM is

$$y(t_i) \approx \sum_j f_j(t_{ij}), \tag{2.3}$$

which t_{ij} is the value of the jth factor. For time-series analysis, it is essential to apply nonlinear trends to account for the periodicity and variance in human behavior, seasonality, and time-sensitive trends. Prophet is based on the GAM, and it is a regression model for inferring time-series data [27]. Prophet is

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t, \tag{2.4}$$

and the equation is a linear regression if $y(\cdot)$ are all linear functions. $g(t)$ is a non-periodic trend modeling function for time-series data, $s(t)$ is a periodic function for seasonally changing data, $h(t)$ is a function that accounts for the effects of holidays, and ϵ_t is any idiosyncratic change that is not accommodated by the model. In Prophet, data are normally distributed as a parametric assumption.

2.2.3 Inference of Machine Learning

Recurrent neural networks (RNNs) offer a machine-learning-based means to investigate time-series data. An RNN is a neural network with a recursive structure. The output of a neural network can use other neural networks as inputs. For time-series data, one way to improve the accuracy of inferences is to consider the input data as part of a series and not independent of each other. The characteristic feature of the RNN is that the output of a hidden unit can use the output of the last layer in a general neural network. One of the well-known RNNs is the long short-term memory (LSTM) architecture. Use of LSTM enables calculation of long-term time-series data in short time [28].

Chapter 3

Basics

In this chapter, we explain the main learning algorithms used in our proposed probabilistic model (i.e., APREP-S), namely multi-class Bayesian logistic regression, hidden Markov model, and k-Shape. We also introduce programming by example (PBE), which constitutes the concept of learning in APREP-S.

3.1 Multi-class Logistic Regression

Multi-class logistic regression is a linear model for classification that takes an input vector $\boldsymbol{x}$ and assigns it to one of D discrete classes. The decision surfaces (i.e., boundaries of decision regions) are linear functions of the input vector $\boldsymbol{x}$ and are defined by $(D-1)$-dimensional hyperplanes within the D-dimensional input space. In APREP-S, we use Bayesian logistic regression, which is a Bayesian treatment of logistic regression, to select one of D pre-defined imputation methods according to the features of the imputation areas.

3.1.1 Bayesian Logistic Regression

Let us consider N input data $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N\}$ as the observations, and N output data $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_N\}$ as the inferences, where $\boldsymbol{x}_n \in \mathbb{R}^M$, $\boldsymbol{y}_n \in \{0,1\}^D$, and $\sum_{d=1}^{D} y_{n,d} = 1$. Here, we assume that $\boldsymbol{Y}$ are generated from the categorical distribution as follows:

$$p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{W}) = \prod_{n=1}^{N} p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{W}) = \prod_{n=1}^{N} \mathrm{Cat}(\boldsymbol{y}_n|f(\boldsymbol{W}, \boldsymbol{x}_n)). \tag{3.1}$$

The categorical distribution is the generalization of the Bernoulli distribution extended to K possible categories. Let $\boldsymbol{\pi} = (\pi_1, ..., \pi_K)^T$ be the parameters of the distribution.

The categorical distribution is expressed such that

$$\mathrm{Cat}(\boldsymbol{s}|\boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{s_k} \tag{3.2}$$

where the multidimensional vector $\boldsymbol{s}_k \in \{0,1\}^K$ satisfies $\sum_{k=1}^{K} \boldsymbol{s}_k = 1$, and the model parameter $\pi_k \in (0,1)$ satisfies $\sum_{k=1}^{K} \pi_k = 1$. The matrix $\boldsymbol{W} \in \mathbb{R}^{M \times D}$ in Eq. (3.1) is the model parameter. Here, we assume the Gaussian prior for each element $w_{m,d}$ of $\boldsymbol{W}$ as follows:

$$p(\boldsymbol{W}) = \prod_{m=1}^{M} \prod_{d=1}^{D} \mathcal{N}(w_{m.d}|0.\lambda^{-1}). \tag{3.3}$$

The softmax function in a D-dimensional space $\mathbb{R}^D$ is used for the nonlinear function $f(\cdot)$. For each dimension d, the function is defined as

$$f_d(\boldsymbol{W}, \boldsymbol{x}_n) = \frac{\exp(\boldsymbol{W}_{:,d}^T \boldsymbol{x}_n)}{\sum_{d'=1}^{D} \exp(\boldsymbol{W}_{:,d'}^T \boldsymbol{x}_n)} \tag{3.4}$$

where $\boldsymbol{W}_{:,d} \in \mathbb{R}^M$ is the d-th column vector of matrix $\boldsymbol{W}$.

The goal in this logistic regression is to obtain the posterior distribution over $\boldsymbol{W}$ when the training dataset $\{\boldsymbol{X}, \boldsymbol{Y}\}$ is given, and to infer the output value $\boldsymbol{y}_*$ when the new input data $\boldsymbol{x}_*$ are given. The posterior probabilities can be expressed using Bayes' theorem:

$$p(\boldsymbol{W}|\boldsymbol{Y}, \boldsymbol{X}) = \frac{p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{W})p(\boldsymbol{W})}{p(\boldsymbol{Y}|\boldsymbol{X})}. \tag{3.5}$$

However, it is generally impossible to calculate the posterior distribution over $\boldsymbol{W}$ analytically because of the nonlinearity derived from the softmax function in $p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{W})$. Therefore, we need approximation techniques, such as function approximation and sampling methods.

Subsequently, by using the obtained approximate posterior distribution, we calculate the inference distribution of $\boldsymbol{y}_*$ for the new input data $\boldsymbol{x}_*$ as follows:

$$p(\boldsymbol{y}_*|\boldsymbol{Y}, \boldsymbol{x}_*, \boldsymbol{X}) = \int p(\boldsymbol{y}_*|\boldsymbol{x}_*, \boldsymbol{W})p(\boldsymbol{W}|\boldsymbol{Y}, \boldsymbol{X})d\boldsymbol{W}. \tag{3.6}$$

It is also impossible to calculate the integral analytically because of the nonlinear function in $p(\boldsymbol{y}_*|\boldsymbol{x}_*, \boldsymbol{W})$. We need to infer $\boldsymbol{y}_*$ by using a further approximation. We show an inference example using a simple Monte Carlo method. This method draws L samples of parameters $\boldsymbol{W}^{(1)}, ..., \boldsymbol{W}^{(L)}$ from the approximate posterior distribution, and obtains the expectation of $\boldsymbol{y}_*$ as follows:

$$\langle \boldsymbol{y}_* \rangle \approx \frac{1}{L} \sum_{l=1}^{L} f(\boldsymbol{x}_*, \boldsymbol{W}^{(l)}). \tag{3.7}$$

3.1.2 Approximate Inference

As mentioned in the previous section, we need some approximation techniques for Bayesian learning. There are two main methods for this purpose. The first is variational Bayes approximation, where mathematically tractable approximate distributions are used for the true posterior. The second is sampling approximation, where samples of latent variables and parameters are employed through sampling techniques, such as the Markov-chain Monte Carlo (MCMC) algorithm.

Variational Bayes inference Regarding function approximation, we introduce variational Bayes inference. It approximates the posterior $p(\boldsymbol{W}|\boldsymbol{Y},\boldsymbol{X})$ by a variational approximation distribution $q(\boldsymbol{W},\boldsymbol{Z})$, where $\boldsymbol{Z}=\{\boldsymbol{z}_1,...,\boldsymbol{z}_n\}$ are the latent variables corresponding to the input data $\boldsymbol{X}$. Here, the calculation of the posterior $p(\boldsymbol{W}|\boldsymbol{Y},\boldsymbol{X})$ in Eq. (3.5) is converted into an optimization problem consisting in minimizing the KL divergence between $q(\boldsymbol{W},\boldsymbol{Z})$ and $p(\boldsymbol{W}|\boldsymbol{Y},\boldsymbol{X})$ as follows:

$$q^*(\boldsymbol{W},\boldsymbol{Z}) = \underset{q(\boldsymbol{W},\boldsymbol{Z})}{\operatorname{argmin}} \ \mathrm{KL}[q(\boldsymbol{W},\boldsymbol{Z}) \parallel p(\boldsymbol{W}|\boldsymbol{Y},\boldsymbol{X})]. \tag{3.8}$$

To address this optimization problem, we consider the relationships between the log-likelihood of data and the KL-divergence as follows:

$$\log p(\boldsymbol{Y},\boldsymbol{X}) = F[q(\boldsymbol{W},\boldsymbol{Z})] + \mathrm{KL}[q(\boldsymbol{W},\boldsymbol{Z}) \parallel p(\boldsymbol{W}|\boldsymbol{Y},\boldsymbol{X})], \tag{3.9}$$

$$F[q(\boldsymbol{W},\boldsymbol{Z})] = \sum_{\boldsymbol{Z}} \int q(\boldsymbol{W},\boldsymbol{Z}) \log \frac{p(\boldsymbol{Y},\boldsymbol{X},\boldsymbol{W},\boldsymbol{Z})}{q(\boldsymbol{W},\boldsymbol{Z})} d\boldsymbol{W}. \tag{3.10}$$

This means that the minimization of the KL divergence is equivalent to the maximization of $F[q(\boldsymbol{W},\boldsymbol{Z})]$ with respect to $q(\boldsymbol{W},\boldsymbol{Z})$ because the log-likelihood, $\log p(\boldsymbol{Y},\boldsymbol{X})$, is constant with respect to $q(\boldsymbol{W},\boldsymbol{Z})$. Therefore, we solve the optimization problem given by

$$q^*(\boldsymbol{W},\boldsymbol{Z}) = \underset{q(\boldsymbol{W},\boldsymbol{Z})}{\operatorname{argmax}} F[q(\boldsymbol{W},\boldsymbol{Z})]. \tag{3.11}$$

We assume

$$q(\boldsymbol{W},\boldsymbol{Z}) = q(\boldsymbol{W})q(\boldsymbol{Z}) = [\prod_{j} q(\boldsymbol{z}_j)]q(\boldsymbol{W}), \tag{3.12}$$

and obtain the update equations by taking the functional derivatives of $F[q(\boldsymbol{W},\boldsymbol{Z})]$ with respect to $\{q(\boldsymbol{z}_j)\}$ and $q(\boldsymbol{W})$. Then, the update formulas are as follows:

$$q(\boldsymbol{z}_j=\boldsymbol{k}) \propto \exp \int q(\boldsymbol{W}) \log p(\boldsymbol{y}_j,\boldsymbol{x}_j,\boldsymbol{z}_j=\boldsymbol{k},\boldsymbol{W}) d\boldsymbol{W} \tag{3.13}$$

$$q(\boldsymbol{W}) \propto p(\boldsymbol{W}) \exp \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \log p(\boldsymbol{Y},\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{W}). \tag{3.14}$$

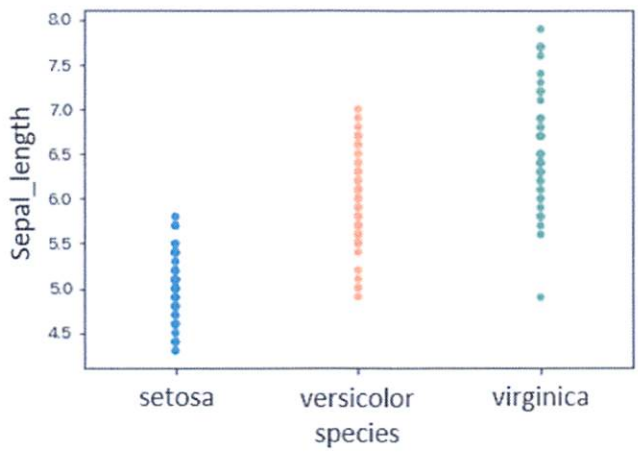

Fig 3.1. *Sepal length* distribution of iris types in the dataset.

Gibbs sampling Concerning the sampling methods, we introduce Gibbs sampling, which is an MCMC algorithm and a special case of the Metropolis-Hastings algorithm. Gibbs sampling generates a sequence of samples from the joint probability distribution of multivariate random variables.

Let us consider the joint distribution $p(\boldsymbol{Z}) = p(\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_n)$ over which sampling is intended. According to Gibbs sampling, we replace the variable $\boldsymbol{z}_i$ with a value drawn from the distribution $p(\boldsymbol{z}_i|\boldsymbol{Z}^{-i})$ that is conditional on the current values of the other variables, where $\boldsymbol{Z}^{-i}$ denotes $\{\boldsymbol{z}_1, ..., \boldsymbol{z}_{i-1}, \boldsymbol{z}_{i+1}, ..., \boldsymbol{z}_n\}$.

3.1.3 Classification Example

As a classification example, we use the Iris DataSet in the UCI repository of machine learning databases [29]. This dataset contains three classes of 50 instances each, where each class refers to a type of iris plant (*Setosa*, *Versicolour*, and *Virginicca*). Each plant type has four feature values, namely *sepal length*, *sepal width*, *petal length*, and *petal width*. The visualization results of the *sepal length* distribution for each type is shown in Fig. 3.1.

Here, we use a linear function $\alpha + \beta \boldsymbol{x}_i$ as an input to $f(\cdot)$ in Eq. (3.1), where α and β are generated from the Gaussian prior. We conducted a numerical experiment based on MCMC using PyMC, which is a programming package for Python that allows users to fit Bayesian models using a variety of numerical methods, including MCMC. Fig. 3.2 shows the inference results for α and β; Table. 3.1 shows their expectations.

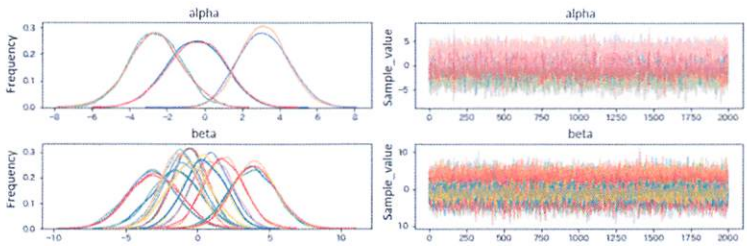

Fig 3.2. Inference result for α and β.

Table 3.1. Expectations of α and β.

parameter		*Setosa*	*Versicolour*	*Virginicca*
α		-0.43	3.16	-2.63
β	*sepal length*	1.49	1.05	0.53
	sepal width	1.77	-0.63	-1.31
	petal length	-3.20	-0.61	3.86
	petal width	-2.99	-0.99	3.96

3.2 Hidden Markov Models

The hidden Markov model (HMM) is widely used in a variety of fields for modeling data sequences, such as speech recognition, natural language modeling, online handwriting recognition, and for analysis of biological sequences such as DNA. Concerning APREP-S, HMM is used for clustering imputation areas into groups with respect to the features of those areas.

Let us consider a sequence of discrete symbols $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N\}$, where observation $\boldsymbol{x}_n$ is generated by a discrete hidden state $\boldsymbol{z}_n$, and the sequence of hidden states $\boldsymbol{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_N\}$ is generated by a first-order Markov process. The joint distribution for this model is given by

$$p(\boldsymbol{X}, \boldsymbol{Z}) = p(\boldsymbol{z}_1) \left[\prod_{n=2}^{N} p(\boldsymbol{z}_n | \boldsymbol{z}_{n-1}) \right] \prod_{n=1}^{N} p(\boldsymbol{x}_n | \boldsymbol{z}_n). \tag{3.15}$$

Each probability distribution is parameterized by $\theta = \{\pi, \boldsymbol{A}, \phi\}$ as follows:
The probability of the first hidden state is

$$p(\boldsymbol{z}_1 | \pi) = \prod_{k=1}^{K} \pi_k^{z_{1,k}} \tag{3.16}$$

where $\pi = \{\pi_k\}$, $\sum_k^K \pi_k = 1$, and the number of states (i.e., clusters) is denoted by K.

The probability of transitioning from state z_{i-1} to state z_i is

$$p(z_n|z_{n-1}, A) = \prod_{k=1}^{K}\prod_{j=1}^{K} A_{jk}^{z_{n-1,j}z_{n,k}} \tag{3.17}$$

where $A = \{A_{jk}\}$ and $\sum_k A_{jk} = 1$. The emission probabilities for each symbol at each state is

$$p(x_n|z_n, \phi) = \prod_{k=1}^{K}\prod_{v=1}^{V} \phi_{kv}^{z_{n,k}x_{n,v}} \tag{3.18}$$

where $\phi = \{\phi_{kv}\}$, $\sum_v \phi_{kv} = 1$, and the number of symbols is denoted by V.

The joint probability distribution over both latent and observed variables is then given by

$$p(X, Z|\theta) = p(z_1|\pi)\left[\prod_{n=2}^{N} p(z_n|z_{n-1}, A)\right]\prod_{n=1}^{N} p(x_n|z_n, \phi). \tag{3.19}$$

3.3 k-Shape

k-Shape is an algorithm for time-series clustering [30]. We use it to search for similar sequences of sensing data in the experimental parts of this thesis. It calculates the centroid of clusters and compares it with each time-series. Then, the data are classified into the closest cluster, and the centroid is updated when the new time-series data arrive. The iteration is repeated until the algorithm converges (e.g., there is no change in cluster memberships).

The algorithm treats observations in time-series data as independent attributes. In general, we consider the invariance of data before clustering, e.g., amplitude scaling, time-shifting, data length scaling, and occlusion. Among these, k-Shape focuses on the invariance of amplitude scaling and time shifting. Concerning the similarity of data in clustering, this algorithm uses cross-correlation with the normalized data as a distance measure. It is called shape-based distance (SBD), which is a domain-independent approach.

Let us consider the similarity of two sequences $x = (x_1, ..., x_m)$ and $y = (y_1, ..., y_m)$. SBD is expressed as

$$\mathrm{SBD}(x, y) = 1 - \max_{\omega}\left(\frac{CC_\omega(x, y)}{\sqrt{R_0(x, x) \cdot R_0(y, y)}}\right), \tag{3.20}$$

where $CC_\omega(x, y) = (c_1, ..., c_\omega)$ is the cross-correlation sequence with length $2m - 1$, defined as

$$CC_\omega(x, y) = R_{\omega-m}(x, y), \quad \omega \in \{1, 2, ..., 2m - 1\}, \tag{3.21}$$

Algorithm 1. k-Shape algorithm [30]

INPUT: X is an n-by-m matrix containing n time-series of length m are initially z-normalized. k is the number of clusters to produce.

OUTPUT: IDX is an n-by-1 vector containing the assignment of n time-series to k clusters (initialized randomly). C is a k-by-m matrix containing k centroids of length m (initialized as vectors with all zeros)

```
iter ← 0, IDX' ← []
while IDX! = IDX' and iter < 100 do
    IDX' ← IDX
    // Refinement step
    for j ← 1 to k do
        X' ← []
        for i ← 1 to n do
            if IDX(i) = j then
                X' ← [X'; X(i)]
            end if
        end for
        C(j) ← ShapeExtraction(X', C(j))
    end for
    // Assignment step
    for i ← 1 to n do
        mindist ← ∞
        for j ← 1 to k do
            [dist, x'] ← SBD(C(j), X(i))
            if dist < mindist then
                mindist ← dist
                IDX(i) ← j
            end if
        end for
    end for
    iter ← iter + 1
end while
```

and $R_{\omega-m}(\boldsymbol{x}, \boldsymbol{y})$ is computed as

$$R_k(\boldsymbol{x}, \boldsymbol{y}) = \begin{cases} \sum_{l=1}^{m-k} x_{l+k} \cdot y_l, & k \geq 0 \\ R_{-k}(\boldsymbol{y}, \boldsymbol{x}), & k < 0. \end{cases} \tag{3.22}$$

The k-Shape procedure is shown in Algorithm 1. We rewrote the algorithm based on Algorithms 2 and 3 in the literature [30].

3.4 Programming by Example

Programming by example (PBE), also termed demonstrational programming, constitutes the concept of learning in APREP-S. In PBE, a system attempts to infer a program exclusively from input and output examples by searching for a composition of base functions [31]. PBE is an end-user development technique to teach a new behavior to a computer by demonstrating action on concrete examples. The system records user actions and infers a generalized program that can be used on new examples. In APREP-S, a set of base functions corresponds to the set of imputation methods, and a program corresponds to our probabilistic model. There are currently many software products based on the PBE approach [32][33][13][12]. In this section, we introduce the PBE model reported in the literature [34].

Probability model Let $\mathcal{S}$ denote a set of strings, and let $f \in \mathcal{S}^{\mathcal{S}}$ denote the target function that maps strings to strings. In the inference phase, the user provides a system input $z := (x, \bar{x}, \bar{y}) \in \mathcal{S}^3$, where x represents the data to be processed, and $(\bar{x}, \bar{y})$ is an example of input-output pair. The goal is to find $f(\cdot)$ such that $\bar{y} = f(\bar{x})$ for each $f \in \mathcal{F}(z)$, where $\mathcal{F}(z) \subseteq \mathcal{S}^{\mathcal{S}}$ is the set of consistent functions for z. To accomplish this goal, we rank the elements in $\mathcal{F}$ using the probability model $\Pr[f|z;\theta]$; θ denotes the parameters learnt during the training phase using a corpus of T training quadruples, $\{(z^{(t)}, y^{(t)})\}_{t=1}^{T}$ with $y^{(t)} \in \mathcal{S}$, which is the correct output on $x^{(t)}$. The system chooses θ that maximizes the likelihood on $\Pr[f^{(1)}, ..., f^{(T)}|z^{(1)}, ..., z^{(T)}; \theta]$.

For $\Pr[f|z;\theta]$, the system uses probabilistic context-free grammar (PCFG). It is defined by a set of non-terminal symbols $\mathcal{V}$, terminal symbols, and rules $\mathcal{R}$. Each rule $r \in \mathcal{R}$ has an associated probability $\Pr[r|z;\theta]$. Using this, the probability of any program $f(\cdot)$ is the probability of its constituent rules $\mathcal{R}_f$ as follows:

$$\Pr[f|z;\theta] = \Pr[\mathcal{R}_f|z;\theta] = \prod_{r \in \mathcal{R}_f} \Pr[r|z;\theta]. \tag{3.23}$$

To connect features with rules, the system uses clues. A clue is a function $c : \mathcal{S}^3 \to 2^{\mathcal{R}}$ that states which subset of rules in $\mathcal{R}$ may be relevant for each z. Suppose the system has n clues $c_1, c_2, ..., c_n$, and let $\mathcal{R}_z = \bigcup_{i=1}^{n} c_i(z)$ be the set of instance-specific rules with respect to z. For each rule $r \in \mathcal{R}_z$,

To connect features with rules, the system uses clues. A clue is a function $c : \mathcal{S}^3 \to 2^{\mathcal{R}}$ that states which subset of rules in $\mathcal{R}$ may be relevant for each z. Suppose the system has n clues $c_1, c_2, ..., c_n$, and let $\mathcal{R}_z = \bigcup_{i=1}^{n} c_i(z)$ be the set of instance-specific

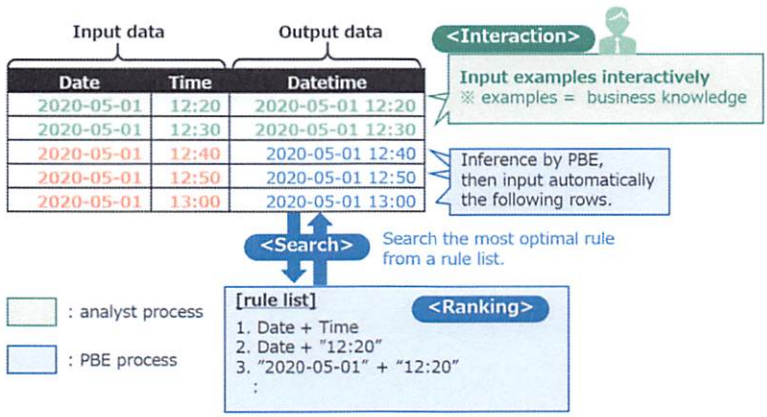

Fig 3.3. Input-output example for a sample case.

rules with respect to z. For each rule $r \in \mathcal{R}_z$,

$$\Pr[r|z;\theta] = \frac{1}{Z_{\mathrm{LHS}(r)}} \exp\left(\sum_{i:r\in c_i(z)} \theta_i\right) \tag{3.24}$$

where $\mathrm{LHS}(r) \in \mathcal{V}$ denotes the non-terminal appearing of r; for each $V \in \mathcal{V}$, the normalizer Z_V is

$$Z_V = \sum_{r\in\mathcal{R}_z:\mathrm{LHS}(r)=V} \exp\left(\sum_{i:r\in c_i(z)} \theta_i\right). \tag{3.25}$$

This is a log-linear model where each clue has a weight e^{θ}.

Training phase During the training time, θ is learnt through the training examples z. We assume that each example $z^{(t)}$ is annotated with the correct program $f^{(t)}$. In this case, we choose θ to minimize the negative log-likelihood of the data with a regularization term:

$$\theta = \underset{\theta'\in\mathbb{R}^n}{\mathrm{argmin}}\{-\log\Pr[f^{(t)}|z^{(t)};\theta'] + \lambda\Omega(\theta')\} \tag{3.26}$$

where $\Omega(\theta')$ is the ℓ_2 norm, and $\lambda > 0$ is the regularization strength (i.e., hyperparameter). If $f^{(t)}$ consists of rules $r_1^{(t)}, r_2^{(t)}, ..., r_{k(t)}^{(t)}$, then

$$\log\Pr[f^{(t)}|z^{(t)};\theta] = \sum_{k=1}^{k=k^{(t)}} \log(Z_{\mathrm{LHS}(r_k^{(t)})}) - \sum_{i:r_k^{(t)}\in c_i(z^{(t)})} \theta_i. \tag{3.27}$$

The parameters θ are optimized by gradient descent.

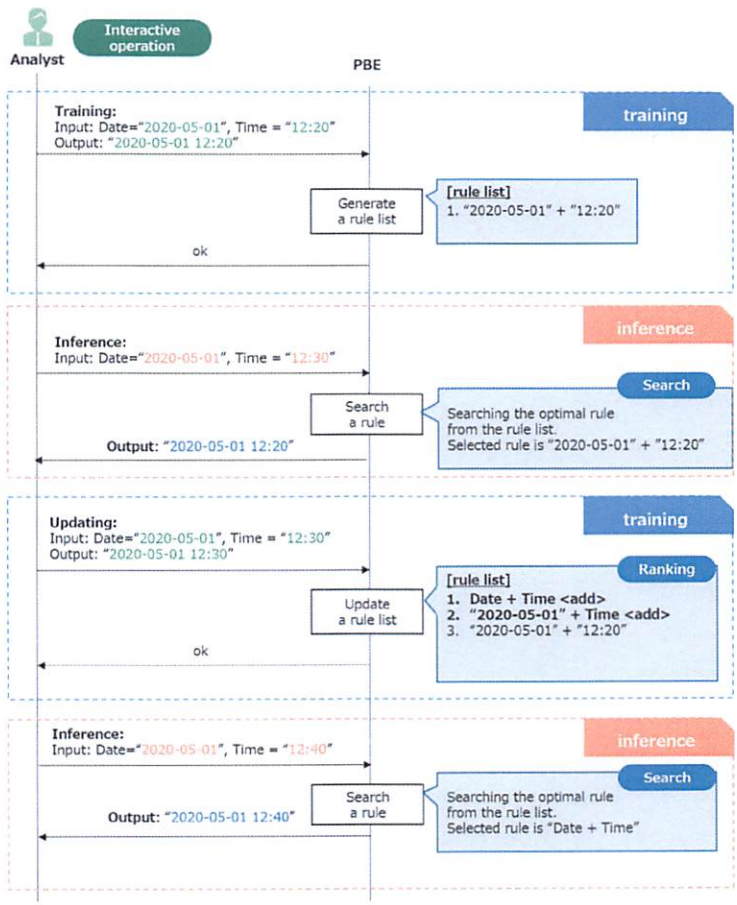

Fig 3.4. Sequence diagram for the sample case. ©2020 IEEE in literature [6]

Inference phase During the inference time, we input $z = (x, \bar{x}, \bar{y})$, $\{c_1, c_2, ..., c_n\}$ and $\theta \in \mathbb{R}^n$ to the system, and infer the most likely program $\hat{f}$ that explains the data under certain PCFG. The procedure is as follows:

1. Evaluate each clue on z under $\mathcal{R}_z$.
2. Assign probabilities to these rules via Eq. (3.24).
3. Enumerate over PCFG in order of decreasing probability, and return the first discovered $\hat{f}$.

Example of PBE As an example of PBE, consider the sample case shown in Fig. 3.3. Here, some input-output pairs are entered into the rows, and then the system infers the appropriate output for some other inputs using the pre-trained PBE probability model. The training and inference processes are iterated and conducted interactively. The details are as follows:

- We enter *Data*=2020-05-01 and *Time*=12:20 as the input and *Datetime*=2020-05-01 12:20 as the output.
- The system generates the translation rule, 2020-05-01 + 12:20, according to the input and adds it into the rule set.
- Then, we enter *Data*=2020-05-01 and *Time*=12:30 as the input. At that time, the system selects the highest rank rule, 2020-05-01 + 12:20 in the rule set, and recommends 2020-05-01 12:20 as the output, but this is not wanted according to our intentions.
- Therefore, we enter again *Data*=2020-05-01 and *Time*=12:30 as the input and *Datetime*=2020-05-01 12:30 as the output.
- As a result, the new rule *Date* + *Time* adds to the rule set of the system. This rule is also used in the subsequent rows.

The whole sequence diagram of the procedure is shown in Fig. 3.4.

Chapter 4

Proposed Framework

In this chapter, we describe the proposed framework for data analysis. The relation in this study is shown in Fig. 1.3.

We propose a data-mining framework termed "automated pre-processing for data mining (APREP-DM)." APREP-DM involves automating steps of the pre-processing of sensor data, including common data-cleaning tasks such as detecting outliers and handling missing data. APREP-DM is based on CRISP-DM, as described in Section 2.1.2. Because CRISP-DM does not depend on specific products, it not only has steps for analysts to treat datasets but its framework defines business understanding as a pre-requirement step to pre-processing. Three iterations are used in APREP-DM: business understanding and data understanding, pre-processing and modeling, and business understanding and evaluation. We evaluate APREP-DM from two perspectives: 1) considering pre-processing in a scenario-based evaluation, assuming pedestrian trajectory tracking, and 2) comparing APREP-DM with the other familiar frameworks at four different points: adding data, business understanding, small iteration, and outlier detection. We conclude that APREP-DM is suitable for analyzing sensor data.

4.1 Overview

We can analyze not only data on existing system, but also integrated data from sensors and wearable devices, making it possible to analyze large amounts of diverse data from multiple perspectives. However, importing raw data directly into the calculations performed by analysis tools such as machine learning models does not provide highly accurate results because of outliers and missing data, mismatched unit and device specifications, and ambiguity within the data. Therefore, pre-processing is required. In particular, data coming from the sensor over the network may be delayed or not received. The delayed data can be outliers, and the nonreceived data become missing

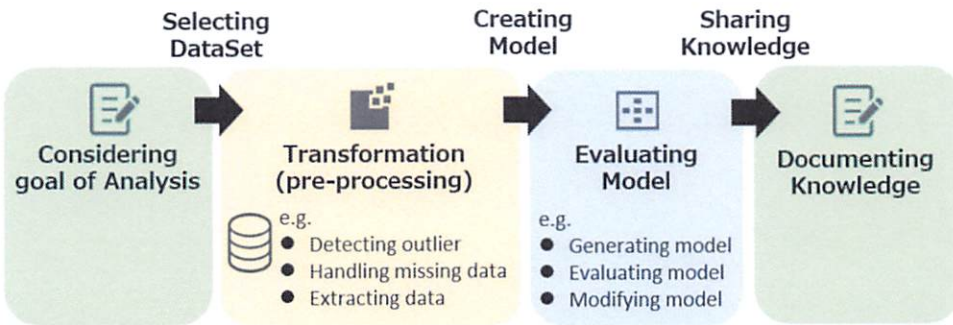

Fig 4.1. Data-mining workflow. Green denotes an interaction with the analyst, yellow pre-processing, and blue the machine learning model. ©2019 IEEE in literature [2]

data. Therefore, analysts themselves must check the outliers and missing data and modify them in some methods.

In earlier studies, frameworks have been proposed for conducting data-mining tasks, covering the full process cycle, starting with the beginning of a project, to model maintenance [35]. A general overview is presented in Fig. 4.1. Data analysts initially consider the goal of the project and then select data from the stored dataset. Then, they transform the data for pre-processing the selected data. Next, they generate and evaluate the machine learning model using transformed data. Finally, they share the documentation about this project and machine learning model as knowledge. The pre-processing step, also called transformation, is the most time-consuming step because of 1) the large quantity and variety of data, 2) the diversification of methods for data analysis, and 3) The many pre-processing tasks required. As mentioned in Section 1.1, pre-processing uses 80% of the resources of the data-mining framework [10]. The well-known data-mining frameworks are described in Section 2.1: KDD, CRISP-DM, SEMMA, and ASUM-DM. KDD can employ small iterations between mining processes, but the processes are complex. CRISP-DM can reveal the priority and criteria of the analysis project clearly, but it is affected by outliers. SEMMA can employ trial and error easily, but it does not consider business understanding nor make use of shared knowledge. ASUM-DM can integrate processes for easy iteration, but it is affected by outliers. Moreover, none of these frameworks mention which process can be run automatically or not. We therefore sought to reduce the pre-processing time by automating parts of the data-mining process.

We here propose a new data mining framework for defining the automatically task of the pre-processing steps in sensor-data analyses. We categorized the tasks of the pre-processing step, whether the task is a common or not in this data analysis, then we

define the common step as an automatically step. By increasing automatically step in a data mining flow, the manual tasks of the analyst decrease. The goal of this chapter are as follows:

- We determine whether or not pre-processing tasks can be automated.
- We verify the effectiveness of APREP-DM in sensor-data analysis using a scenario-based and qualitative evaluation.

4.2 Design

We focus on the pre-processing step in the data-mining framework and propose a new framework involving the automated step APREP-DM. APREP-DM has an automated sub-step and a nonautomated sub-step for pre-processing. The automated sub-step handles tasks that are based on statistics, clustering or classifying items that depend on the goal and criteria for the analysis, such as detecting outliers and handling missing data. We call this sub-step a "common process for pre-processing." The nonautomated substep is a trial-and-error step for finding the most suitable models of the analysis goal, such as extracting data and reconstructing the dataset. We refer to this as the "other process for pre-processing." An overview of APREP-DM is shown in Fig. 4.2. The common process for pre-processing requires a business goal and criteria for priority and success. If the analyst is unable to decide some conditions for outliers from a business understanding aspect, APREP-DM cannot run the automated step. Therefore, APREP-DM defines the business understanding step before the pre-processing step. The two pre-processing sub-steps (3-1) and (3-2) are not performed simultaneously. They are sequential steps: First, the common process for pre-processing is performed, and next the other process for pre-processing is performed. The specific workflow is as follows:

1. Business understanding: clarify the goal of analysis and the criteria for priority and success. The goal and criteria help define outliers and handle missing data. The output is the dataset for the analysis.
2. Data understanding: understand the stored data used in the project and select data for analysis. This includes the listing task of data and sampling data if necessary. The analyst creates a target dataset from the original dataset.
3. Pre-processing

   (3-1) Common process for pre-processing: transform the target dataset for detecting outliers and handling missing data based on the goal and the criteria

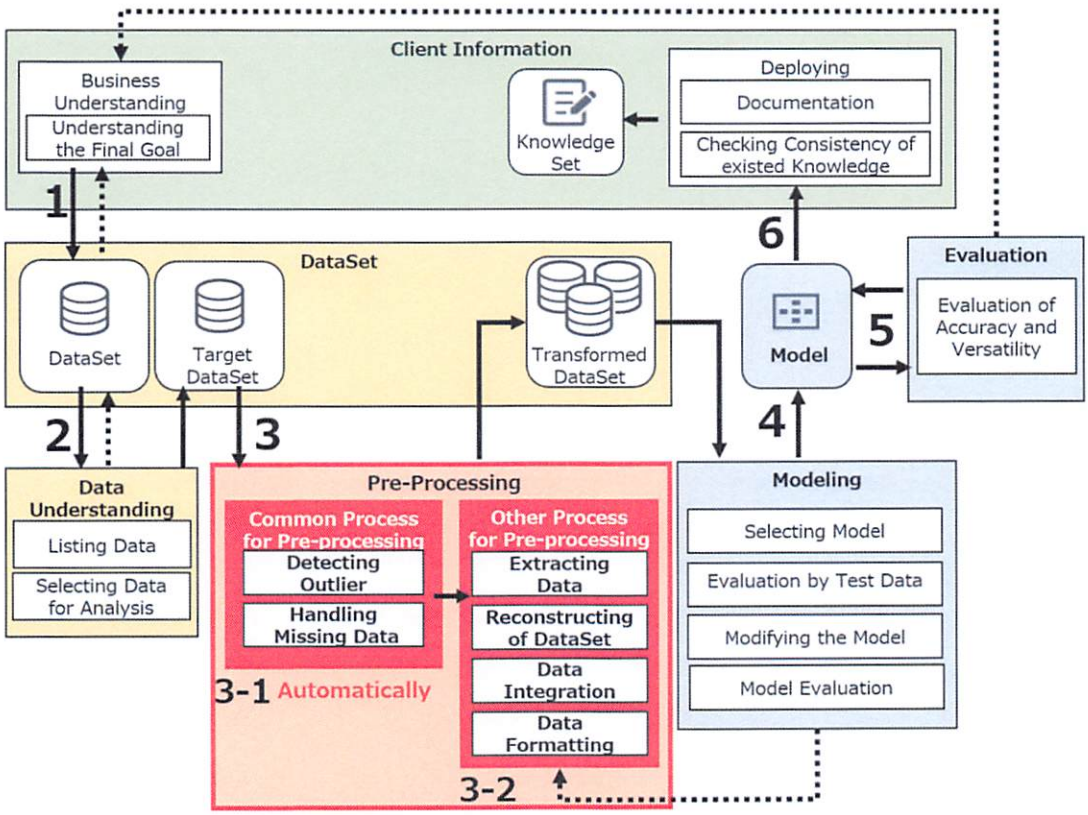

Fig 4.2. Overview of the APREP-DM framework. The red portion denotes proposed steps. ©2019 IEEE in literature [2]

of the analysis. This sub-step can be run automatically by the statistical method or by clustering.

(3-2) Other process for pre-processing: create the transformed dataset for the modeling step by trial and error. This sub-step is a manual step performed by the analyst. It involves extracting data, reconstructing the dataset, data integration, and transforming the data format.

4. Modeling: generate and evaluate a model using the transformed dataset. This transformed dataset can be used as training data for a model. It defines a machine learning model, for example, a decision tree or neural network. If the defined model is not suitable for the analysis, the analyst tries another machine learning model or recreates transformed data in the other process for the pre-processing step. The output of this step is a model for the analysis.

5. Evaluation: evaluate the generated model in the modeling step for accuracy and versatility using applications. If the result of the evaluation exhibits no problems, then the analyst defines this model as the analysis model. If the model is not suitable for analysis, the analyst reconsiders the goal and criteria for the analysis in the business understanding step.

6. Deployment: summarize the process of the data mining and knowledge sharing of the model by, e.g., providing documentation.

Although APREP-DM is based on CRISP-DM as mentioned above, it can detect outliers during pre-processing. This is needed because outliers impact the statistics significantly and should therefore be detected early. However, APREP-DM only detects outliers; it does not remove them. Outliers can be important for detecting anomalies or unexpected behavior. Therefore, the analyst can select whether or not to address the outliers and remove them if necessary.

4.3 Evaluation

We evaluate the proposed framework APREP-DM from two perspectives:

1. Scenario-based evaluation, assuming pedestrian trajectory tracking using sensor data.

2. Qualitative evaluation, comparing APREP-DM with other familiar frameworks from four aspects.

We clarify pre-processing processes that can automatically be used for scenario evaluation. Then, we verify that APREP-DM is a suitable framework for the analysis of sensor data.

4.3.1 Scenario-based Evaluation

We evaluate a scenario that analyzes customer behavior using multiple three-dimensional (3D) range-imaging sensors:

> The system is aimed at predicting customer behavior in a shopping mall with three exits, using data describing where individual customers exit the mall depending on their point of entry. Based on this result, we deliver suitable coupons for the shopping mall by a push function on a mobile application.

The goal of this analysis is to obtain the highest possibility arriving exit based on

customer features. For example, the result is that a person entering from exit 1 will go to exit 2 by inputting features such as velocity and weather. In this evaluation, the training data are one-day data of weekdays and weekends, where Wednesday, 24th October 2012, is a representative weekday, and Sunday, 28th October 2012, is a representative weekend or holiday. The inference data are Wednesday, 14th November 2012, and Sunday, 18th November 2012.

Dataset

We use a dataset comprising multiple range images obtained using 3D sensors [7] and a meteorological dataset [36]. The sensor data involve outliers and missing data. The walking trajectories of shopping mall customers were monitored and the data were gathered for 92 days over approximately one year, between 9:40 and 20:20 every Wednesday and Sunday from October 2012 to November 2013. The data from Wednesday, 24th October 2012, are approximately 17 million rows, and the data from Sunday, 28th October 2012, are approximately 41 million rows. The locations of customers were measured continuously at a rate of 10–40 Hz using multiple 3D range-image sensors. The shopping mall has three exits: 1) to a ferry terminal, 2) to a train station, shops, and offices via escalators and elevators, and 3) to a commercial and catering area on the eastern side.

In this evaluation, each exit of the x axis and y axis are defined as square areas. The area of exit 1 is $-45000 < x < -30000$ and $-8000 < y < 0$, the area of exit 2 is $-10000 < x < 10000$ and $9000 < y < 15000$, the area of exit 3 is $38000 < x < 50000$ and $-30000 < y < -15000$. A summary is given in Table 4.1. An image of the data is shown in Fig. 4.3. Fig. 4.3 (a) is the floor map of the ATC shopping mall, Fig. 4.3 (b) is constructed by using all of the one-day data on 24th October 2012, and Fig. 4.3 (c) is constructed by using the first 10 people as data points on the same day. As shown in Fig. 4.3 (c), the trajectory data do not connect one exit with another. The sensor data are stored in CSV format, and they contain the following columns: UNIX time, *person_id*, positions *pos_x* and *pos_y*, *hight* [mm], *velocity* [mm/s], *body_angle* of motion [rad], and *facing_angle* [rad]. A summary is given in Table 4.2.

The meteorological dataset comprises information from Osaka, where the shopping mall is located. Data were downloaded for the period from October 2012 to November 2013 and include *date*, *temperature* [° C], *rainfall*, *windspeed*, and *weather* parameters. A summary is given in Table 4.3. *weather* has 15 meteorological types, such as sun, cloud, or rain.

Table 4.1. Exiting the shopping mall.

Exit	Leads to	x axis	y axis
1	ferry terminal	$-45000 < x < -30000$	$-8000 < y < 0$
2	train station, shops, and offices	$-10000 < x < 10000$	$9000 < y < 15000$
3	commercial and catering area	$38000 < x < 50000$	$-30000 < y < -15000$

Table 4.2. Units of the sensor data.

Name	Unit
UNIX time	-
person_id	-
pos_x	mm
pos_y	mm
hight	mm
velocity	mm/s
body_angle	rad
facing_angle	rad

Table 4.3. Units of meteorological data.

Name	Unit
date	*year/month/date h:min:sec*
temperature	°C
rainfall	mm
windspeed	m/s
weather	-

Procedure

Machine learning was performed using a support vector machine (SVM) [37], currently one of the most commonly used pattern-recognition models. We used a sklearn library for the SVM. The SVM obtains seven columns (*weekday*, *from*, *mean_velocity*, *mean_rainfall*, *mean_temperature*, *mean_windspeed*, and *weather*) as input data, and one column (*to*) as output data. *weekday* is a flag specifying whether or not the data are recorded on a weekday, *from* is an entered exit number, and *weather* is the identification (id) of the meteorological type. *mean_velocity* is the mean of the velocity for each person from the exit to the exit; *mean_rainfall*, *mean_temperature*, and *mean_windspeed* are mean values for each data as well as *mean_velocity*. In addition, *to* is the exit number of the exit in the shopping mall.

In this evaluation, we chose two days of data as the training data: Wednesday, 24th October 2012, and Sunday, 28th October 2012. We chose two days of data as the inference data: Wednesday, 28th November 2012, and Sunday, 2nd December 2012. Both sets are in the fourth week of the month. Some *person_id* values are -1, but these are outliers because *person_id* is a seven-digit number. In addition, *person_id* values that do not exist for more than 3 s are configured as outliers because the time is too short to define a person having entered the shopping mall. Three seconds means walking approximately 3 m, which is calculated by the mean of the data on 24th October, 942.428 mm/s. The sensor operates at 10-40 Hz, and so we can extract *person_id* for more than 100 rows. The meteorological dataset contained some missing data in

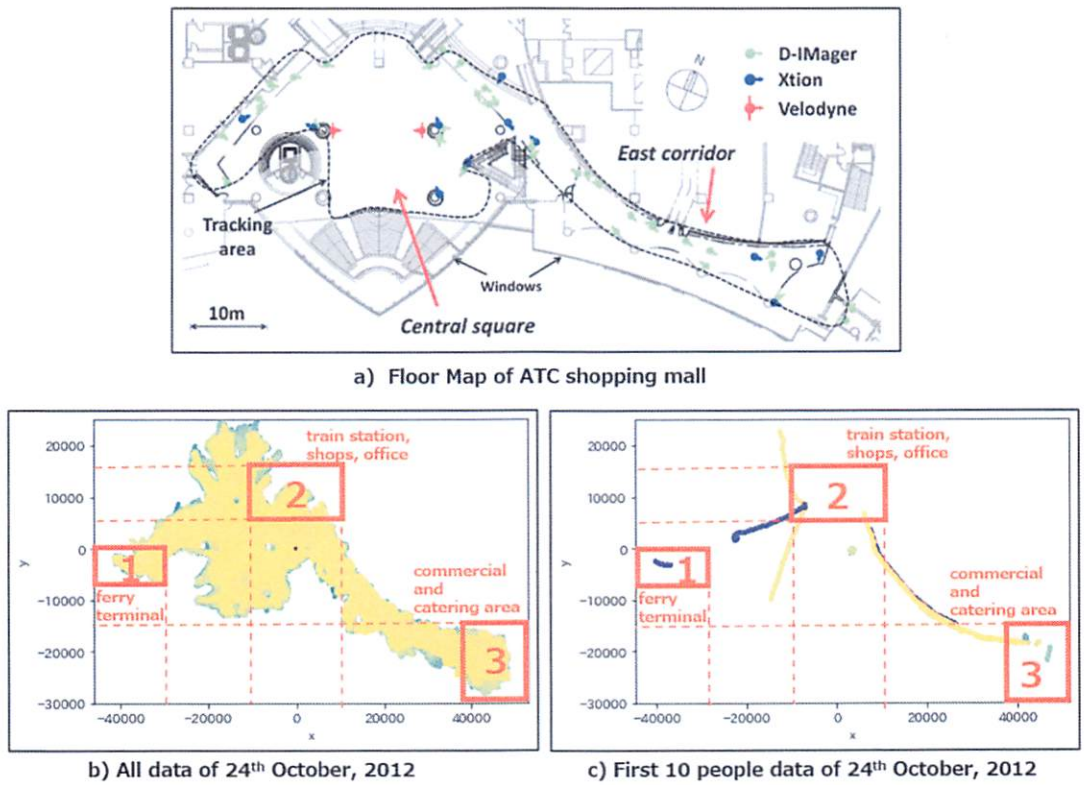

Fig 4.3. Map of the shopping mall using 3D range-imaging sensor data. (a) ©2019 IEEE in literature [7]. In (b) and (c), the red frames denote the area of each exit. (b) is redrawn based on the Fig. 6 in literature [2].

the *weather* column because *weather* data are measured every 3 h in the meteorological dataset. Therefore, we extracted nonmissing data from the meteorological data. Moreover, the required data are connected data from one exit to another exit. The exit can be the same as the entrance. Therefore, data for which both the first location and the last location are not positioned in the exit area, as listed in Table 4.1, are deleted.

The specific pre-processing processes are as follows. We use Talend [38], which is one of the popular data transforming tools. To generate training data and inference data for the SVM, the sensor and meteorological data are joined. The pre-processing tasks are shown in Fig. 4.4. The tasks indicated by the blue arrows can be processed automatically in APREP-DM. We clarify the automated step in APREP-DM, and it can process automatically if we have business understanding before pre-processing. Because this evaluation is not an anomaly analysis, the outliers can be deleted. This is decided in the business understanding step. We can delete the rows that have

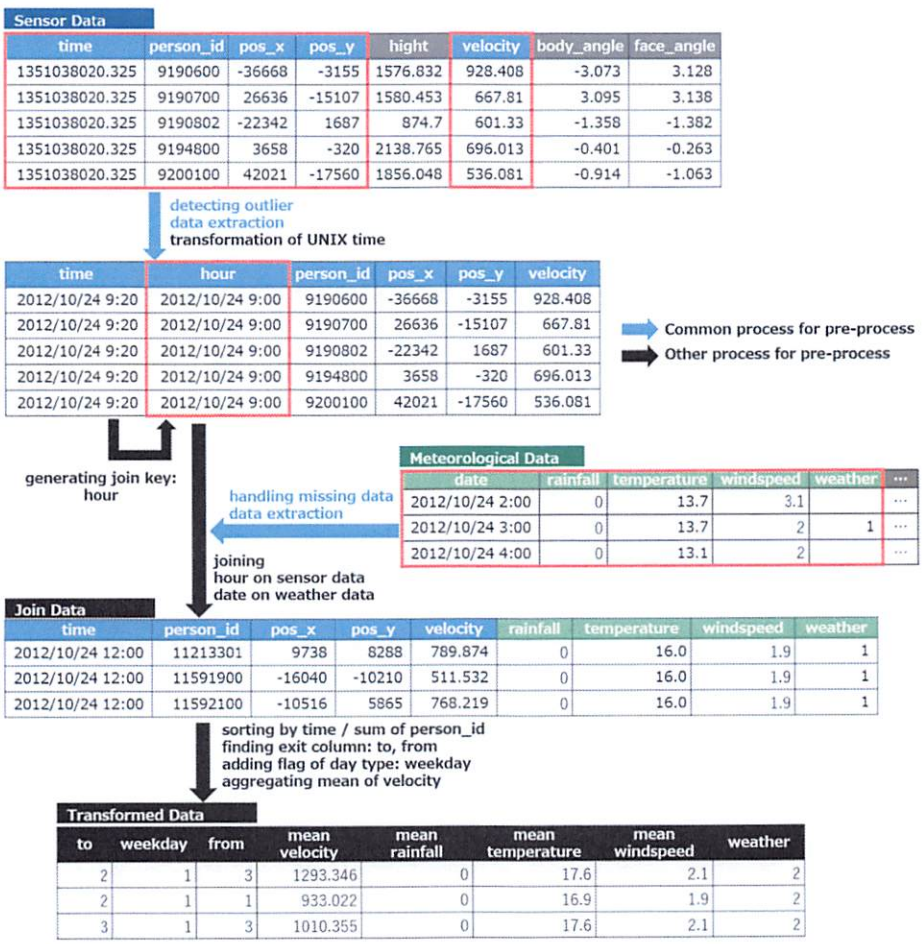

Fig 4.4. Pre-processing in the scenario-based evaluation.

$person\_id = -1$ or for which the data are too short. Therefore, we delete the outliers in the common process for pre-processing. In addition, in the common process for pre-processing, we handle the missing data because some rows of *weather* are missing. These two steps are automatically process as transforming rules.

The data-mining process can be summarized as follows:

1. Business understanding: the goal of this analysis is to obtain the highest possibility arriving exit by using the customer features. The features are weekday or not, the entering exit, mean of the velocity, and weather. Because this scenario of analysis is not anomaly analysis, we deleted the outlier rows from the dataset and extracted the required columns of data.

2. Data understanding (format and contents of items): the features can be generated from the sensor and meteorological datasets. Some *person_id* values are −1, and some of them are of too short a duration. The meteorological data are measured every 3 h, and there are different formats of data between the sensor dataset and meteorological dataset. Moreover, we selected the required data from both datasets.

3. Pre-processing:

   (a) Common process for pre-processing: detecting outliers and handling missing data.

      (i) Transforming the dataset to take into account outliers about *person_id* in the sensor dataset. In this evaluation, outliers are deleted.

      (ii) Handling missing data about *weather* in the meteorological dataset. In this evaluation, deleting missing data using the Listwise method.

   (b) Other process for pre-processing: data integration, aggregation, and transforming data for analysis. The output is transformed data and it is used as input data into the SVF.

      (i) Creating a join key named *hour* in the sensor dataset to join two different datasets.

      (ii) Aggregating and calculating data for drawing each person's walking trajectory.

      (iii) Generating necessary data. In this evaluation, a weekday flag is generated.

      (iv) Unifying the format *time* in the sensor dataset with *date* in the meteorological dataset.

4. Modeling and evaluation: generating the SVM model using transformed data and evaluation.

Evaluation Result and Discussion

The summary is shown in Table 4.4. After quantitative pre-processing, the transformed training data comprise 3,591 rows generated from the data on Wednesday, 24th October 2012, and Sunday, 28th October 2012. The transformed inference data comprise 3,775 rows generated from the data on Wednesday, 28th November 2012, and Sunday, 2nd December 2012. Both the training data and inference data have decreased numbers of rows, because, in this evaluation, the data are aggregated in *person_id*. However, the number of transformed data was much less than *person_id*. The reason for this is considered to be the use of a list-wise deletion method for handling missing data.

Table 4.4. Number of rows for transformed data.

	Date	Original	*person_id*	Transformed
training data	Wed, 2012/10/24	16,817,749	6,654	1,317
	Sun, 2012/10/28	40,957,069	15,622	2,274
	sum	57,774,818	22,276	3,591
inference data	Wed, 2012/11/28	16,814,061	6,309	1,156
	Sun, 2012/12/2	39,909,590	16,879	2,619
	sum	56,723,651	23,188	3,775

**person_id* means the number of unique ids in the original data.
*Transformed data means the data inputting to the SVM.

4.3.2 Qualitative Evaluation

We evaluated APREP-DM from four aspects in the analysis of sensor data. The features of APREP-DM involve business understanding and an automated pre-processing step. We compared APREP-DM features with earlier data-mining frameworks.

Aspect of Qualitative

The analysis model requires that the input data be suitably integrated by integrating the appropriate columns. Furthermore, it is necessary to detect and remove outliers and to include business understanding when deciding which parts of the pre-processing are to be automated. Finally, we evaluated the iteration scheme to ensure that each step can be easily iterated to improve the accuracy of the analysis model. Therefore, we evaluated the following aspects of the sensor data analysis:

- Adding data: we investigate the ease of adding columns by reconstruction, aggregation, or joining in the middle of the framework.
- Business understanding: we define the goal and criteria of the analysis and clarify the information and processes needed to select the data required for the analysis.
- Small iteration: we iterate data-mining steps flexibility between small range.
- Outlier detection: we detect outliers and, when necessary, remove them.

Based on the above four aspects, we compares APRPE-S with four earlier frameworks: KDD, CRISP-DM, SEMMA, and APREP-DM.

Evaluation Result and Discussion

The results are listed in Table 4.5. APREP-DM is the most suitable method, except from the small iteration aspect. The iteration steps of APREP-DM occur before pre-

Table 4.5. Comparison with earlier frameworks.

Framework name	Adding data	Business understanding	Small iteration	Outlier detection
APREP-DM	+++	+++	++	+++
KDD	+	++	+++	++
CRISP-DM	+++	+++	++	+
SEMMA	+	+	N/A	+++

* The number of + symbols indicates the degree of adequacy.
* N/A means not applicable.

processing, during pre-processing and generation of the model, and after evaluation, while the iteration of KDD is defined in every step. Therefore, KDD is the most suitable framework from the aspect of small iterations. We describe the specific results in the following.

On the addition of data, KDD and SEMMA can reduce the size of the dataset from the original one, while they cannot add any columns. CRISP-DM and APREP-DM involve data integration in the pre-processing step. Therefore, CRISP-DM and APREP-DM can add columns more easily than KDD and SEMMA.

With regard to business understanding, SEMMA does not involve a business understanding step because it does not have any analyst steps. In KDD, CRISP-DM, and APREP-DM, there is a step in which the project goal is decided first. Moreover, CRISP-DM and APREP-DM involve a step for considering the priority of aim and criteria. Therefore, CRISP-DM and APREP-DM can set a more specific goal than KDD.

In terms of small iterations, KDD can iterate any two steps, whereas CRISP-DM and APREP-DM regard one cycle as multiple steps in the data-mining workflow. Therefore, KDD is the smallest and the most flexible iteration step among KDD, CRISP-DM, and APREP-DM. Although SEMMA explains the iteration of data mining as a natural step, it does not have any clear iteration step. Therefore, we consider this is not applicable (N/A) in this evaluation.

For outlier detection, CRISP-DM does not have any step pre-processing step. KDD, SEMMA, and APREP-DM have a process for outliers. Moreover, APREP-DM and SEMMA only detect outliers, while KDD removes outliers. Therefore, we can use APREP-DM or SEMMA for abnormal analysis projects.

4.4 Summary

We evaluated APREP-DM using the scenario-based and qualitative evaluation. In the scenario-based evaluation, we clarified the automated steps and verified the data mining

framework. In the qualitative evaluation, we compared APREP-DM with four well-known frameworks from four aspects. The conclusions of this chapter are as follows:

- Clarifying business understanding first (i.e., defining the goal and the criteria of the analysis) is essential for selecting and transforming data from the dataset.
- There are some steps that can be processed automatically, though the quantity of the data decrease considerably though pre-processing.
- APREP-DM is a well-balanced framework that is suited to analyzing sensor data.

Chapter 5

Proposed Imputation Method

In this chapter, we describe the proposed data imputation method used in the pre-processing stage of data analysis processes. The relation in this study is shown in Fig. 1.3.

We propose "automated pre-processing for sensor data (APREP-S)," which is an imputation method for pre-processing. It uses Bayesian inference for the calculation of the proportion of the likelihood of each imputation model, whereas it adopts a programming by example (PBE) approach to update the APREP-S model through a user interface. It contains the multiple imputation methods, and the method that leads to the optimal method is determined based on the features of the target imputation area. The input is the target imputation data, and the output is the data inferred by APREP-S. The analyst confirms the output data, and if it is necessary to update the APREP-S model, this is performed by the imputation value and appropriateness of each data imputation method. Four experiments are conducted to evaluate APREP-S regarding data periodicity, types of training data, feature flexibility, updating, and generating models. It is demonstrated that APREP-S is an effective imputation method, particularly for sensor data.

5.1 Overview

Data collected by sensors and wearable devices are increasingly being analyzed in autonomous robot behavior analysis, customer trend analysis, and task management in factories. Particularly, depending on network conditions, sensor data may be delayed, timed out, or lost during transmission. Moreover, data acquired by a battery-powered sensor may be lost in the case of battery depletion.

A well-known workflow for data analysis is CRISP-DM, which is described in Section 2.1.2. We proposed "automated pre-processing for data mining (APREP-DM) [2]"

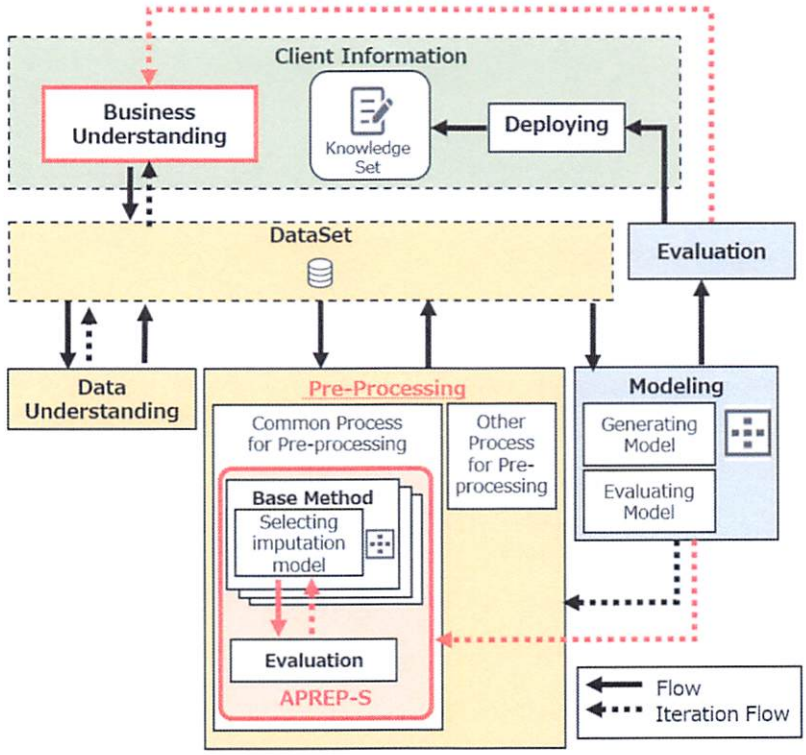

Fig 5.1. Overview of APREP-DM and APREP-S. The drawing is based on overview of APREP-DM. The red frame and red dashed arrows indicate the related flows in APREP-S. ©2020 IEEE in literature [6]

based on CRISP-DM. This method allows the automatic execution of a pre-processing step. In addition, we proposed APREP-S [6] [8] [9] [39] to automate the pre-processing step through the integration of human knowledge by focusing on the corresponding pre-processing step of APREP-DM.

The role of APRPE-S within APREP-DM is shown in Fig. 5.1. In APREP-DM, the pre-processing step comprises a common process to handle missing data and outliers, and another process to transform the form fit of the model. APREP-S is an imputation method applied in the former process. It first generates the base methods of the imputation models in APREP-S, and subsequently selects the optimal imputation model from these methods. The complementary data provided by APREP-S are used to generate the analysis model. If the model requires regeneration, the process flow returns to APREP-S or iterates from the first business understanding step. The ranked PBE rules are the models of an imputation model in APREP-S, and the PBE rank is the likelihood of each imputation model calculated by APREP-S. The examples used

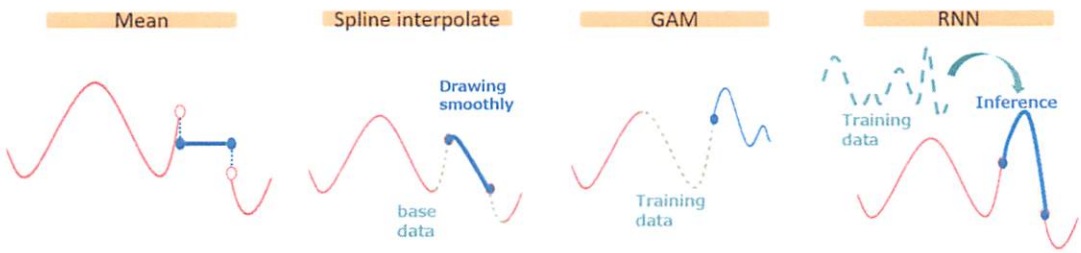

Fig 5.2. Schematic of data imputation.

as input are the features of the APREP-S model. The PBE approach is described in Section 3.4. As the analyst interactively inputs the updated data, the rules that are the ranks of the imputation models are updated. The analyst can generate and update the APREP-S model and obtain complementary data by simply entering a few input-output pairs.

As mentioned in Section 1.4.2, existing imputation methods for handling outliers and missing data can be classified into two categories: manual and automated processes. In the former, the analyst manually operates the necessary tools, whereas in the latter, the analyst firstly needs to generate a machine learning model then maintains. The objective of this study is to integrate manual and automated methods. In the former, customization is easy, but many tasks should be performed by the analyst. By contrast, in the latter, fewer tasks are involved, as the analyst only generates the machine learning model, but customization and maintenance are too difficult for people who are not with IT skill. Because, these methods require IT knowledge. Therefore, we propose an automated method with the customization ability of manual methods. It utilizes existing imputation methods as candidates of the method, such as specifically described in Section 2.2. A schematic of data imputation is shown in Fig. 5.2. If the imputation method uses only the mean of the two data parts, all imputation values are the same in the target imputation area. If the imputation method uses only spline interpolation, it only outputs a smooth curve. However, a time-series analysis, such as a GAM, and a machine learning method, such as an RNN, allows inferring the imputation values in the target imputation area. As these approaches have their own advantages and disadvantages, we indicate how to select the most suitable model from several imputation methods.

In a business environment, an IT engineer is often responsible for generating a common model and distributing it to project sites. Although the best approach would be to assign an IT engineer to every project and every site, this would be difficult because the number of IT engineers is limited [15], as described in Appendix A.

Therefore, we often use a network to connect IT systems, as in Industry 4.0. A cyber-physical system, as in Industry 4.0 [17], would enable a project expert to use machine learning models at the project site because an IT engineer would be able to generate machine learning models and deliver them over the network to other sites. However, IT engineers are unable to timely update and maintain such a model so that it can be adjusted to reflect the different features of each site, such as site climate, employee behavior, and project rules. Therefore, a project expert should be able to maintain the model without assistance from an IT engineer. Hence, we need to find a way to enable the model to be updated by project experts. An example based on a factory is shown in Fig. 5.3, which includes the added features of the model in Fig. 1.4. An IT engineer first generates the flexibility model at the model-generating site, and subsequently a project expert updates that model to adjust the features of each site at the project site. Specifically, the initial model is generated by using all the features A, B, C, D, and E, as input, which may be used at each project site. This model is distributed to each project site, and subsequently the project expert updates the model by using site-specific features as input, for example, A, B, C, and D in Factory A.

Our objectives in this chapter are as follows.

- To propose an imputation method for outliers and missing data based on machine learning integrated with human knowledge using a PBE approach to reduce the required analysis resources.
- To evaluate APREP-S, which selects several imputation models according to the features of the target imputation area by integrating human knowledge.
- To compare APREP-S with existing imputation methods to assess its effectiveness against outliers and missing data in terms of accuracy of imputation and the similar trend of the original data.

5.2 Probability Formulation

We begin with a formal discussion of the proposed approach. We first define the related terms. Then, we describe the APREP-S model.

5.2.1 Terminology

A summary of related terms is provided in Table 5.1. As shown in Fig. 5.3, an IT engineer in the IT department is often responsible for generating the common model and distributing it to project sites. Here, the term a model-generating site is a site (e.g., IT department) where a common model is generated, and a project site is a site

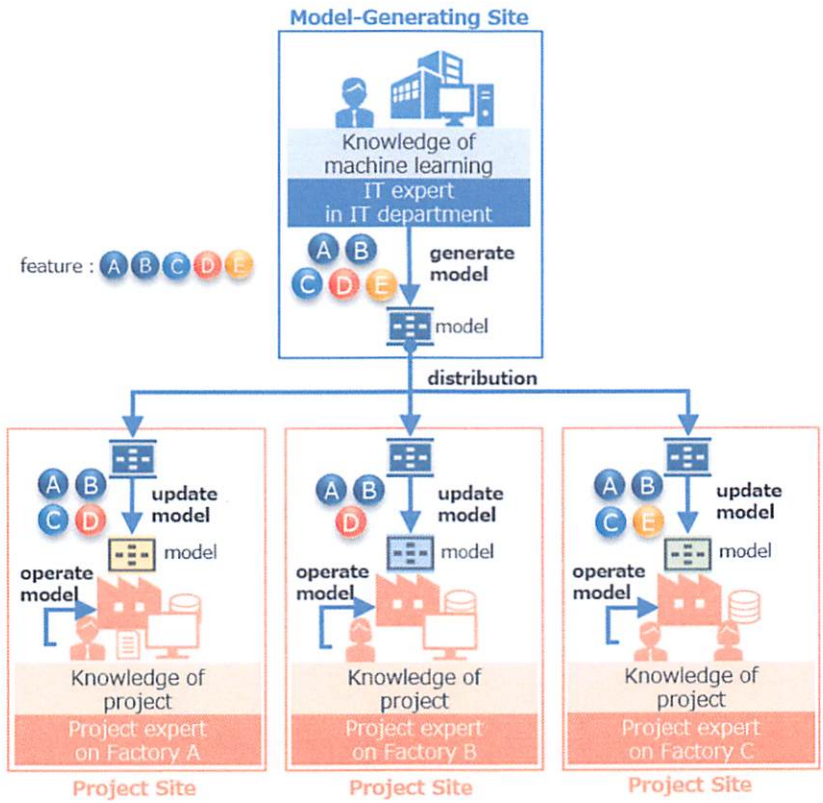

Fig 5.3. Example of model-generating site and project sites.

where the model is operated and updated. The term IT engineer refers to an engineer with machine learning skills at the model-generating site, and project expert refers to an experienced person with project knowledge at the project site. The project expert has less machine learning knowledge than the IT engineer.

In addition, the term target imputation data refers to data with missing values and outliers. The inference process will be applied to these data by APREP-S. The term imputation method refers to methods such as spline interpolation and LSTM, imputation model is a model generated by an imputation method, and target imputation area is a range that requires continuous imputation in the target imputation data. The target imputation area has certain sub-areas, and APREP-S can select the imputation model for each such sub-area. An imputation value is the value calculated by the imputation model. These concepts are visualized in Fig. 5.4. The horizontal axis represents time, which is discretized because it is assumed that sensor data are measured at a constant rate, such as 30 Hz. These imputation values are inferred by APREP-S. In the example

Table 5.1. Term definitions for APREP-S.

Term	Description
Model-generating site	Site (e.g., IT department), where a common model is generated.
Project site	Site where a site-specific model is operated and updated.
IT engineer	Engineer with machine learning skills at the model-generating site.
Project expert	Experienced person with project knowledge at the project site.
Imputation method	Imputation method(s) in APREP-S.
Imputation model	Model(s) generated from an imputation method. (one of the candidates that APREP-S selects).
Target imputation data	Data with missing values and outliers.
Target imputation area	Range requiring continuous imputation.
Target imputation sub-area	Sub-area in target imputation area.
Imputation value	Value calculated by the imputation model in target imputation sub-area.

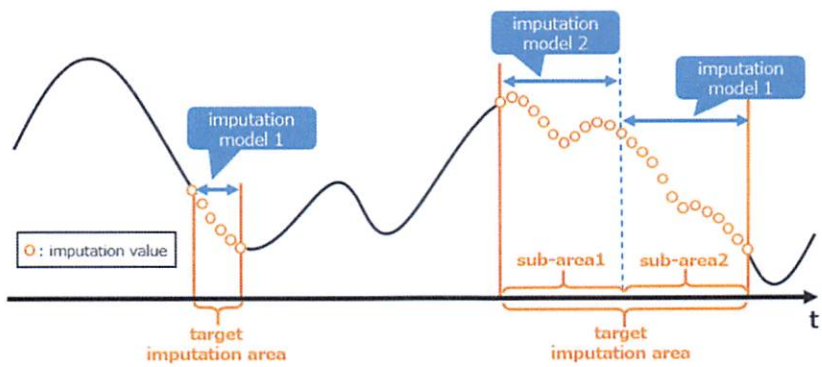

Fig 5.4. Schematic of imputation.

shown in Fig. 5.4, the first target imputation area is indicated by imputation model 1, and the second target imputation has two sub-areas indicated by imputation model 2 and imputation model 1.

5.2.2 Probability Model

The APREP-S model infers the optimal imputation models in each imputation area of the target data. The APREP-S model is generated and updated to improve accuracy, and the imputation values for the target imputation data are inferred.

The input-output data of APREP-S for training and inferring are shown in Fig. 5.5. To generate the APREP-S model, the training data TR_A are received and transformed into the input form of the APREP-S model, which is a pair of features $\boldsymbol{X} =$

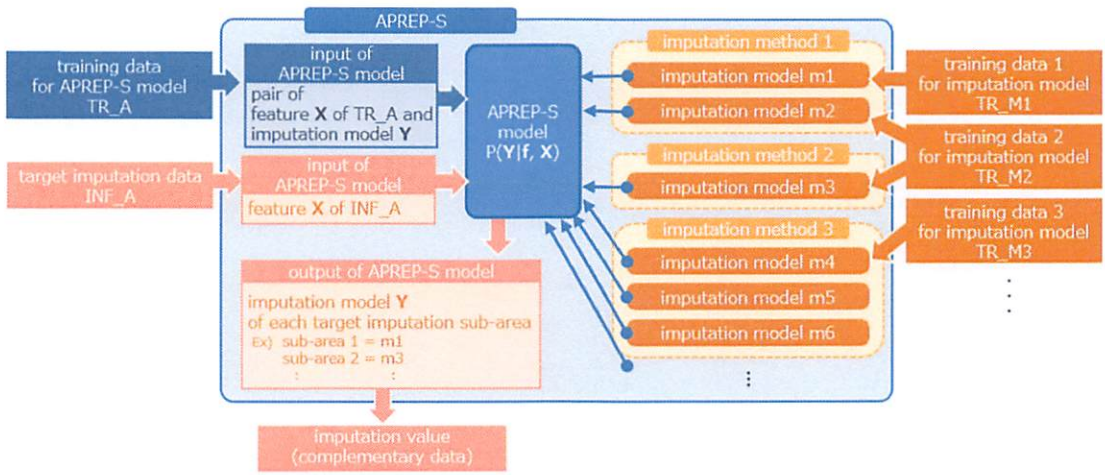

Fig 5.5. Input-output data of APREP-S model for training and inferring.: Blue areas indicate model training, yellow areas indicate candidate imputation models, and orange areas indicate inference by the APREP-S model.

$\{\boldsymbol{x}_1, ..., \boldsymbol{x}_D\}$ and imputation models $\boldsymbol{Y} = \{y_1, ..., y_D\}$, where $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^D$. D is the size of the sum of the target imputation area. $\boldsymbol{x}_d \in \mathbb{R}^Q$, where $1 \leq d \leq D$, are normalized features for the APREP-S model, and Q is their number. If the features are time, temperature, and wind speed, the non-normalized $\boldsymbol{x}'_d = [10, 12, 0.1]$ implies that the time is 10:00, the temperature is 12°C, and the wind speed is 0.1m/s, and then $\boldsymbol{x}_d$ has the corresponding normalized values. $\boldsymbol{Y}$ represents the selected imputation models generated depending on TR_A by any rule. $\boldsymbol{Y}$ is a $D \times K$ matrix, where $y \in \mathbb{N}$, and K is the number of imputation models defined in APREP-S. For example, if $D = 2$ and $K = 3$, then if we select model 1 in the first target imputation area, and model 3 in the second target imputation area, then $\boldsymbol{Y} = [1, 3]$.

The APREP-S model infers the probability of the imputation models for each target imputation area using a linear classification model; thus,

$$p(\boldsymbol{Y}|\boldsymbol{f}, \boldsymbol{X}) \tag{5.1}$$

where $\boldsymbol{f}(\cdot)$ is a non-linear function (softmax function [40] [41]), which has two parameters α and β generated from a Gaussian distribution. The size of α is K, and β is a $K \times Q$ matrix, where $K \in \mathbb{N}$ is the number of imputation models. $\boldsymbol{f}(\cdot)$ is

$$\boldsymbol{f}(x_d) = \alpha + \beta\, x_d \quad (1 \leq d \leq D). \tag{5.2}$$

APREP-S trains two parameters: α and β. Here, the APREP-S model is based on Bayesian inference:

$$p(\boldsymbol{f}|\boldsymbol{Y}, \boldsymbol{X}) \propto p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{f})p(\boldsymbol{f}). \tag{5.3}$$

The APREP-S model selects the optimal imputation model $m_k \in \mathcal{M}$ $(1 \leq k \leq K)$, where $\mathcal{M}$ is a set previously generated imputation models. That is, $\mathcal{M}$ are candidates in the APREP-S model. The posterior distribution of each imputation model $p(m_k|\boldsymbol{f})$ is calculated based on Bayesian inference:

$$\begin{aligned} p(m_k|\boldsymbol{f}) &= \frac{p(\boldsymbol{f}|m_k)p(m_k)}{\sum_{i=1}^{K} p(\boldsymbol{f}|m_i)p(m_i)} \\ &= \frac{\exp(\boldsymbol{f}(x_k))}{\sum_{i=1}^{K} \exp(\boldsymbol{f}(x_i))}. \end{aligned} \tag{5.4}$$

As $\mathcal{M}$ is a set of discrete elements, it is a categorical distribution. The likelihood function is

$$C(\mathcal{M}|\boldsymbol{f}) = \prod_{k=1}^{K} (f_k^{u_k}) \tag{5.5}$$

where u_k denotes the probability that the method is m_k. $\sum_k y_k = 1$, and $0 \leq y_k \leq 1$. $p(m_k|\boldsymbol{f})$ is a normalized exponential function because $\sum_{i=1}^{K} p(m_i|\boldsymbol{f}) = \sum_{i=1}^{K} u_i = 1$. Here, the probability of the imputation models $P_y(\mathcal{M}|\boldsymbol{f}, \boldsymbol{X})$ is calculated from $\boldsymbol{f}(\cdot)$(Eq. (5.2)). $P_y(\mathcal{M}|\boldsymbol{f}, \boldsymbol{X})$ is

$$P_y(\mathcal{M}|\boldsymbol{f}, \boldsymbol{X}) = C(\mathcal{M}|\boldsymbol{f}). \tag{5.6}$$

Subsequently, as shown in Fig. 5.5, the target imputation data INF_A are received as input. The output is the pair of the target imputation sub-area and imputation model $\boldsymbol{Y}$. For example, if imputation model 2 is optimal for sub-area 1, and imputation model 1 is optimal for sub-area 2, $\boldsymbol{Y} = [2, 1]$. Then, APREP-S outputs an imputation value as complementary data. A summary of the values related to the APREP-S model is shown in Table 5.2.

5.3 Method Details

APREP-S contains the definitions of multiple imputation models, and it leads to the optimal model can be determined based on the features of the target imputation area. The input is the target imputation data, and the output is the data inferred by APREP-S. The analyst confirms the output data, and if it is necessary to update the APREP-S model, this is performed by comparing the imputation value and appropriateness of each data imputation method. Human knowledge can be input into the APREP-S model using the PBE approach when the model is updated. The ranked PBE rules are the models of the imputation method in APREP-S, and the PBE rank is the likelihood

Table 5.2. Values for APREP-S model.

Value	Description
TR_A	Training data for APREP-S model.
TR_M, TR_M_S	Training data for the imputation models in APREP-S.
$\boldsymbol{X}$	Features of the imputation area for the APREP-S model
$\boldsymbol{x}_d \in \boldsymbol{X}$	Features of each target imputation value for the APREP-S model. the size is Q.
$\boldsymbol{Y}$	Selected imputation models for the APREP-S model.
$\mathcal{M}$	Set of imputation models (discrete elements).
K	Number of elements in M.
D	Size of the sum of the target imputation area, that is, the size of $\boldsymbol{Y}$.
Q	Number of features.

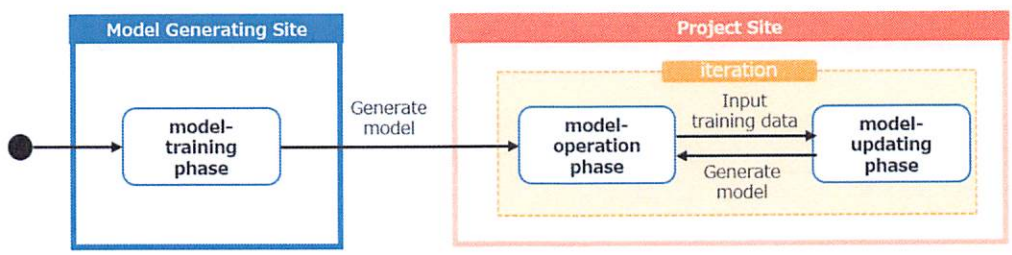

Fig 5.6. Training, operating, and updating phases in APREP-S. ©2020 IEEE in literature [6]

of each imputation model calculated by APREP-S. The examples used as input are the features of the APREP-S model. As the analyst interactively inputs the updated data, the rules that are the ranks of the imputation models are updated. The analyst can generate and update the APREP-S model, and obtain complementary data by simply entering a few input-output pairs.

APREP-S has three main phases, namely, model training, updating, and operating, as shown in Fig. 5.6. The first phase takes place at the model-generating site, and the others the project site. In the model-training phase, the initial APREP-S model is generated. In the model-updating phase, the model is updated. In the model-operating phase, the imputation values are inferred and calculated. As the initial model is generated in the model-training phase, APREP-S iterates the model-operating and the model-updating phase after generating the initial model.

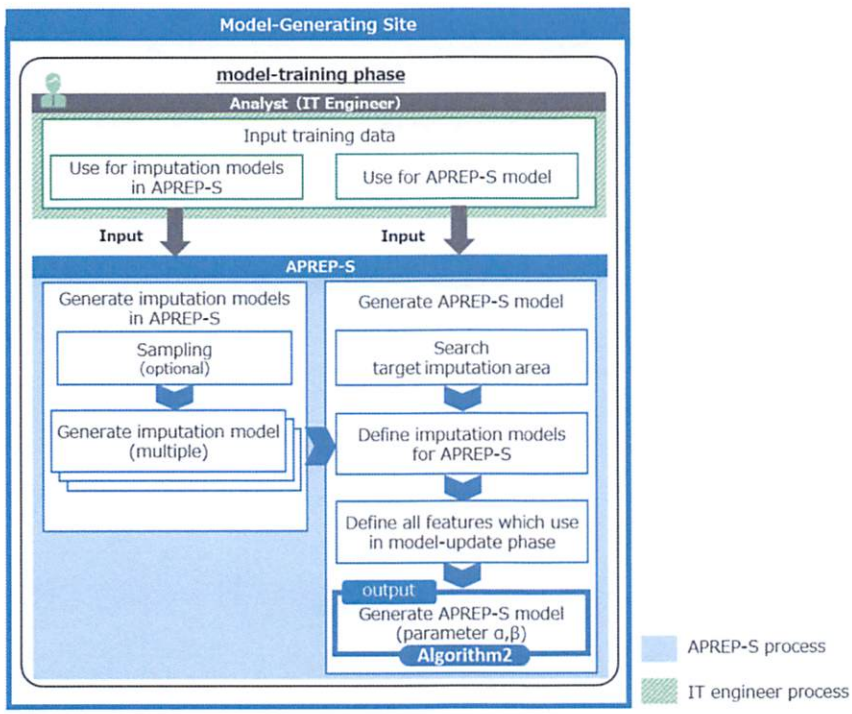

Fig 5.7. Workflow of model-training phase. ©2020 IEEE in literature [6]

5.3.1 Model-Training Phase

In the model-training phase, an APREP-S model is generated. Initially, two types of data are received for the imputation models to be selected and for the APREP-S model. Subsequently, imputation models are generated; these can be selected in the model-operating phase. In parallel, the APREP-S model is generated with inference model parameters α and β. The imputation models are constructed during the generation process for the APREP-S model. The output of the model-training phase is the APREP-S model.

The workflow is shown in Fig. 5.7. A more detailed description is as follows:

1. The analyst inputs training data TR_M and TR_A.
2. APREP-S generates the imputation models.
   (a) If necessary, TR_M is sampled, and TR_M_S is generated.
   (b) Imputation models are generated using TR_M or TR_M_S.
3. The APREP-S model is generated.

Algorithm 2. Generation of APREP-S model (model-training phase).

INPUT: A pair $\boldsymbol{X}$ and $\boldsymbol{Y}$
$\boldsymbol{X}$ is a list of normalized features calculated from TR_A.
$\boldsymbol{Y}$ is the list of the selected imputation models.
D is the size of the target imputation area.
Q is the size of the features $\boldsymbol{x} \in \boldsymbol{X}$.
m_k is an imputation model in $\mathcal{M}$. The size is K.
OUTPUT: APREP-S model $\boldsymbol{f}(x_d) = \alpha + \beta\, x_d \quad (1 \leq d \leq D)$.

$\alpha \leftarrow \mathcal{N}(\mu_\alpha, \sigma_\alpha)$
$\beta \leftarrow \mathcal{N}(\mu_\beta, \sigma_\beta)$
for $d \leftarrow 1$ *to* D **do**
  for $k \leftarrow 1$ *to* K **do**
    for $k \leftarrow 1$ *to* Q **do**
      $\boldsymbol{f} = \alpha + \beta\, x_d \leftarrow \alpha, \beta$
      $p(m_k|\boldsymbol{f}) = \exp(\boldsymbol{f}(x_d))/\sum_{i=1}^{K} \exp(\boldsymbol{f}(x_i)) \leftarrow m_k, \boldsymbol{f}$
      $p(\boldsymbol{f}|m_k) \leftarrow \boldsymbol{Y}$
      $C(m_k|\boldsymbol{f}) \leftarrow \boldsymbol{f}, p(\boldsymbol{f}|m_k)$
      $\alpha_t, \beta_t \leftarrow$ sampling with $C(m_k|\boldsymbol{f})$
    end for
  end for
end for
APREP-S model $\boldsymbol{f} \leftarrow \alpha_t, \beta_t$

(a) Searching the target imputation area, that is, detecting outliers and missing data. Then, APREP-S generates the list of the selected imputation model $\boldsymbol{Y} = \{y_1, ..., y_D\}$ using TR_A as training data and any imputation model.

(b) The imputation models $\mathcal{M}$ are defined.

(c) All features $\boldsymbol{X}$ that may be used in the model-update phase are defined.

(d) The APREP-S model $\boldsymbol{f}$ is generated. The algorithm for training the APREP-S model is shown in Algorithm 2.

5.3.2 Model-Operating Phase

In this phase, the imputation values are inferred by the APREP-S model. The analyst first inputs the target imputation data to APREP-S. Subsequently, APREP-S searches the imputation target area from the inference data. Then, APREP-S defines the normalized features $\boldsymbol{X}$, and calculates the likelihood of each imputation model from the APREP-S model generated in the model-training phase, including the parameters α and β. This result is $\boldsymbol{Y}$. Then, APREP-S selects the optimal model by comparing the corresponding likelihoods. If the target data have more than one sub-area, implying

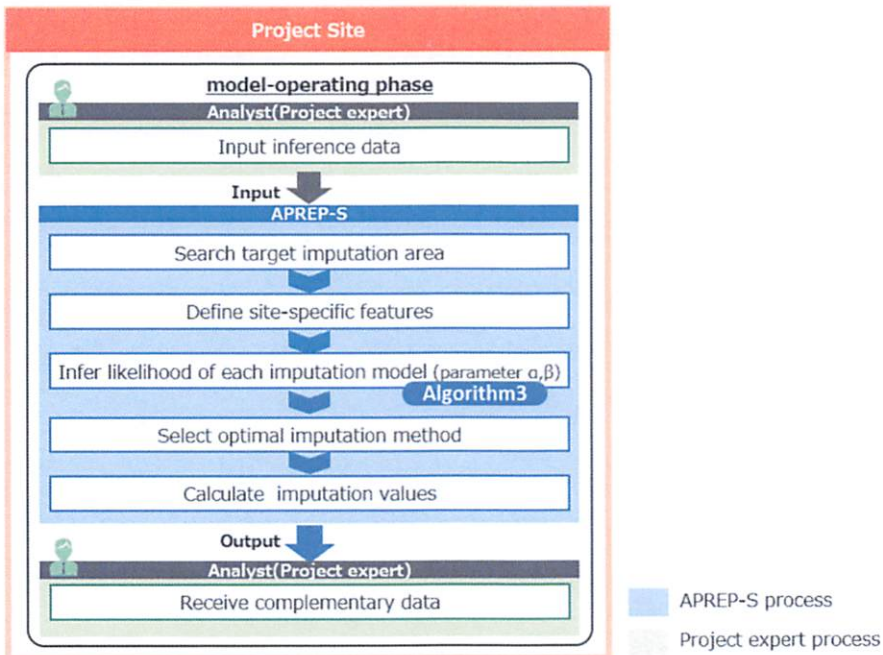

Fig 5.8. Workflow of model-operating phase. ©2020 IEEE in literature [6]

consecutive target imputation values, APREP-S defines the target imputation sub-area, calculates the likelihoods, and selects the optimal imputation models. For example, if APREP-S has three models m_1, m_2, m_3, and inputs x_n as the nth feature in the target imputation area, $\boldsymbol{Y}$ is calculated by using the APREP-S model, for example, $\boldsymbol{P}_y = [0.1, 0.3, 0.6]$. In this case, we select the highest proportion method, thus, APREP-S calculates the nth imputation value by m_3 in the target imputation area. Next, APREP-S calculates the imputation values by using the selected imputation model in each target imputation sub-area. Finally, APREP-S returns the complementary data to the analyst.

The workflow of the model-operating phase is shown in Fig. 5.8, and a more detailed explanation is as follows:

1. The analyst inputs the target imputation data (inference data for the APREP-S model) INF_A.

2. APREP-S searches the target imputation area in INF_A.

3. APREP-S generates the normalized features $\boldsymbol{X}$, that is, site-specific features.

Algorithm 3. Inference likelihood of each imputation model (model-operating phase).

INPUT: $\boldsymbol{X}$ represents normalized features calculated from INF_A.
D is the size of the target imputation area.
Q is the size of feature $\boldsymbol{x} \in \boldsymbol{X}$.
m_k is an imputation model in $\mathcal{M}$. The size is K.
APREP-S model is a generated model in Algorithm 2.
$g(\cdot)$ is any function for selecting the optimal imputation models
OUTPUT: The selected optimal imputation models $\boldsymbol{Y}$

$\boldsymbol{f} \leftarrow$ APREP-S
for $d \leftarrow 1$ *to* D **do**
  for $k \leftarrow 1$ *to* K **do**
    $\mathbb{E}_\alpha[\alpha|m_k] = \sum_q p(\alpha_p|m_k)\alpha_p$
    $\mathbb{E}_\beta[\beta|x_q, m_k] = \sum_q p(\beta_p|m_k, x_q)\beta_p$
    $\alpha_{fix} \leftarrow \mathbb{E}_\alpha[\alpha|m_k]$
    $\boldsymbol{\beta_{fix}} \leftarrow \mathbb{E}_\beta[\beta|x_q, m_k]$
    $\boldsymbol{f} = \alpha + \beta \boldsymbol{x}_d \leftarrow \alpha_{fix}, \beta_{fix}$
    $p(m_k|\boldsymbol{f}) = \exp(\alpha + \beta \boldsymbol{x}_d) / \sum_{k=1}^{K} \exp(\alpha + \beta \boldsymbol{x}_d) \leftarrow \boldsymbol{X}$
    $\boldsymbol{Y} \leftarrow g(p(m_k|\boldsymbol{f}))$
  end for
end for

4. The likelihood of each model imputation model are inferred using the APREP-S model. The related algorithm is shown in Algorithm 3.

   (a) Set two probability parameters α and β; the size of α is K, and β. In the calculation, α and β are defined as fixed values of the mean of each imputation model, denoted by α_{fix} and β_{fix}, respectively. APREP-S calculates the expectations $\mathbb{E}_\alpha$ and $\mathbb{E}_\beta$ of α and β, respectively.

   (b) The APREP-S model $\boldsymbol{f}$ is defined (Eq. (5.2)), with α_{fix} and β_{fix}.

   (c) For each $m_k \in \mathcal{M}$, a posterior probability $p(m_k|\boldsymbol{f})$ is defined (Eq. (5.6)). The APREP model for each method m_k is

$$p(m_k|\boldsymbol{f}) = \frac{\exp(\alpha_{fix} + \beta_{fix} m_k)}{\sum_{i=1}^{I} \exp(\alpha_{fix} + \beta_{fix} m_i)} \tag{5.7}$$

   (d) For each $m_k \in \mathcal{M}$, define a prior probability: $p(\boldsymbol{f}|m)$ from a likelihood function $C(\mathcal{M}|\boldsymbol{f})$ is defined (Eq. (5.5)).

5. The optimal imputation models in each target imputation sub-area is selected. This is the output of the APREP-S model $\boldsymbol{Y}$.

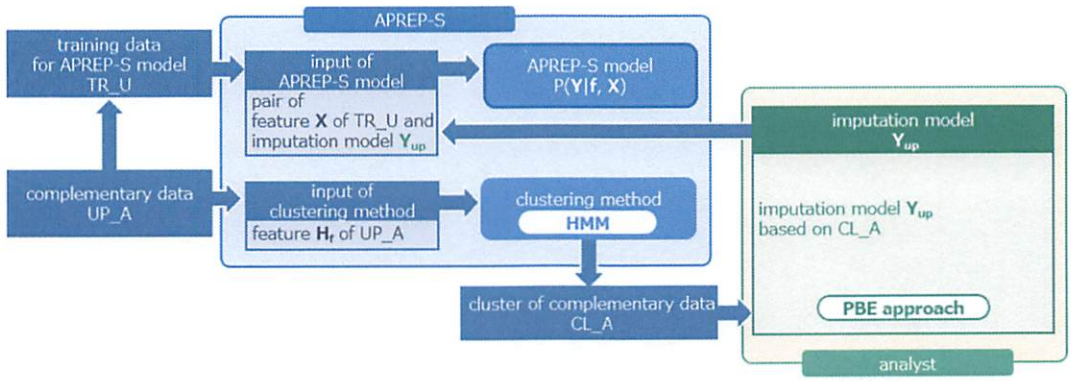

Fig 5.9. Input-output data of the APREP-S model for updating.: The blue areas indicate model training, whereas green areas indicate the generation of new training data by the analyst.

6. The imputation values are calculated using the selected imputation model.

7. The complementary data are output to the analyst.

5.3.3 Model-Updating Phase

In the model-updating phase, the APREP-S model is updated using a clustering method and the PBE approach. The input-output data of the APREP-S model for updating are shown in Fig. 5.9. The analyst first inputs complementary data UP_A to generate the new training data for the APREP-S model update, and the cluster CL_A of UP_A is returned by using the hidden Markov model (HMM) as a clustering method, as described in Section 3.2. This is because the HMM rapidly returns the number of clusters, thereby matching the interaction of the PBE process. Although K-means is a well-known clustering methods is [42], it cannot process time-series data. Therefore, we use the HMM for this purpose. Then, the analyst proceeds to generate training data $\boldsymbol{Y}_{up}$ for updating CL_A using the PBE approach. The APREP-S model is updated using the training data TR_U, which are extracted as complementary data UP_A and $\boldsymbol{Y}_{up}$.

The workflow of the model-operating phase is shown in Fig. 5.10, and it is described in detail as follows.

1. The analyst checks the complementary data handled in the model-operating phase.

2. The received complementary data are classified by the HMM. The algorithm is shown in Algorithm 4.

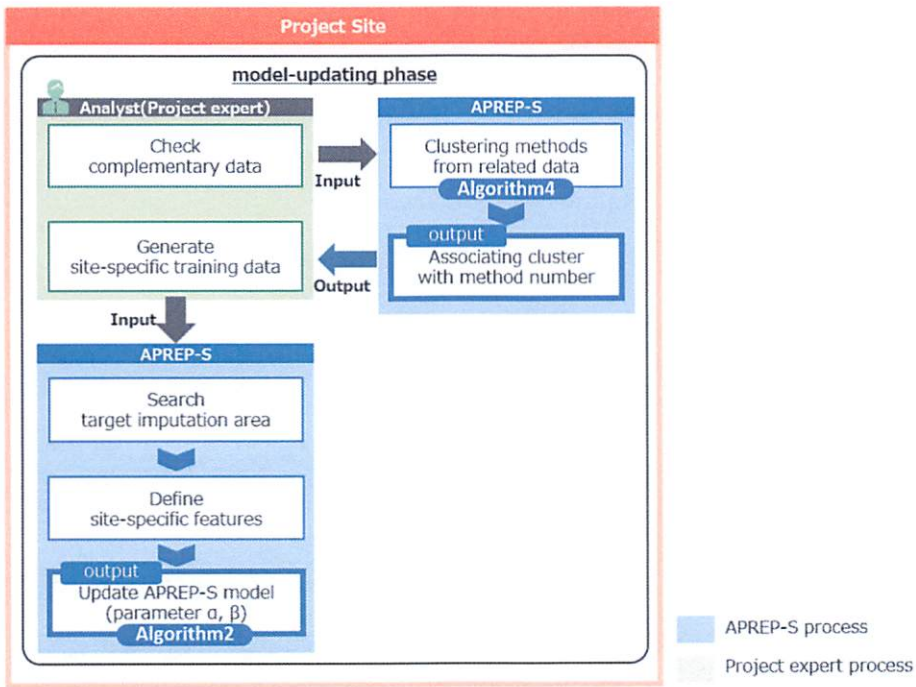

Fig 5.10. Workflow of model-updating phase. ©2020 IEEE in literature [6]

(a) The features $\boldsymbol{H}_f$ of UP_A are generated for the clustering method.

(b) $\boldsymbol{H}_f$ is input to clustering method (HMM). The number of clusters may be arbitrary. The HMM has a sequence of observable variables $\mathbf{X}$, where $\mathbf{X}$ is $\boldsymbol{H}_f$.

(c) APREP-S infers the HMM parameters π, $\mathbf{A}$, and θ, and returns the clustering result CL_A.

3. APREP-S returns CL_A to the analyst.

4. The analyst generates new training data $\boldsymbol{Y}_{up}$ using the PBE approach.

(a) The clusters and the imputation models are associated as site-specific training data $\boldsymbol{Y}_{up}$. The interface for the clustering of the models is shown in Fig. 5.11. The upper part is for inputting the site-specific data into APREP-S. The features are uploaded when the project expert clicks on the upload button. If the project expert has feature data, the project expert uploads all site-specific feature data. Subsequently, the project expert inputs the number of clusters and clicks on the run button. Here, the button of cluster

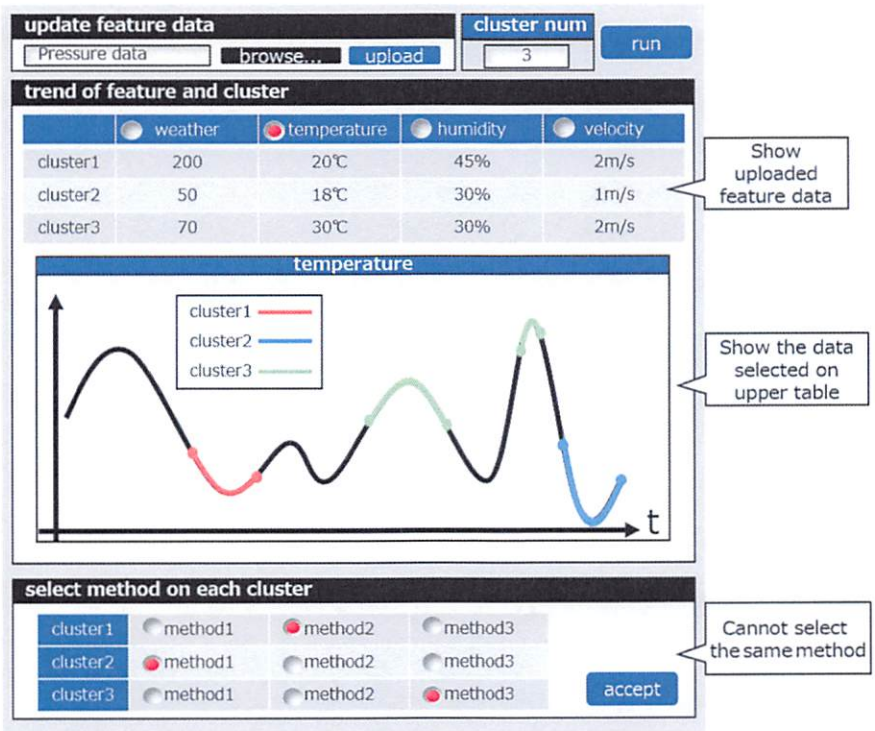

Fig 5.11. Interface for the clustering of imputation models. : The red solid circle next to the name of an item and method indicates that they have been selected. The chart area displays the selected data listed in the table in the upper part. Temperature data are currently selected for display.

num refers to the number of clusters that APREP-S is required to return. The interface shows the trend of the input data and the chart classified by the clustering method. The project expert selects the optimal method for each cluster in the lower part, where these trends are visualized.

(b) Input $\boldsymbol{Y}_{up}$ to APREP-S.

5. $\boldsymbol{Y}_{up}$ and TR_U are received. TR_U is the data with missing values and outliers before the imputation of UP_A.

6. The target imputation area in TR_U is searched.

7. A normalized feature $\boldsymbol{X}$ is generated, that is, site-specific features.

8. The APREP-S model is generated. The algorithm is shown in Algorithm 2, which is the same as in the mode-generating phase.

Algorithm 4. Generating new training data for APREP-S model (model-updating phase).
INPUT: Complementary data UP_A generated in model-operating phase
$\boldsymbol{H}_f$ represents the features of UP_A.
OUTPUT: CL_A is the cluster of complementary data.

1: $\boldsymbol{H}_f \leftarrow$ features (UP_A)
2: $CL\_A \leftarrow$ HMM($\boldsymbol{H}_f$)

5.4 Evaluation

We evaluated APREP-S using both short- and long-term periodicity data as follows: 1) human activity data (short-term periodicity), and 2) temperature and humidity data (long-term periodicity). We used human activity data by assuming such as trajectory analysis or behavioral analysis. In this study, four experiments are conducted, and a summary is shown in Table 5.3.

Experiment 1 is intended to verify whether accuracy improves by selecting a certain imputation method in each target imputation area. We use training data with a similar trend to that of the target imputation data, and evaluate the first imputation accuracy of APREP-S using fixed features, that is, the features in the model-generating site and the project site do not change. The target imputation data have the target imputation positions, that is, the range of all continuous imputations is 1.

Experiment 2 enhances Experiment 1. It is a verification of the accuracy of APREP-S in a target imputation area with a target imputation sub-area and a continuous imputation area. For continuous imputation, we evaluate the effectiveness of machine learning imputation methods. The training data have a similar-trend to that of the target imputation data, but the two datasets are not the identical.

Experiment 3 is intended to verify whether APREP-S is an efficient method in the case of using own-data to train the imputation models. In addition, we evaluate the update process in APREP-S with site-specific features. This experiment is assumed the case that no similarity data is in the target imputation data.

Experiment 4 is intended to verify whether accuracy improves by updating the APREP-S model using site-specific features. We evaluate short- and long-term periodicity data, and similar data to the target imputation data are used for training. In addition, we evaluate the efficiency of generating multiple imputation models from a single imputation method.

Table 5.3. Experiments.

Number	Experimental data	Training data*	Update process	Features**
Experiment 1	long-term periodicity	similar-trend data	none	fixed
Experiment 2	long-term periodicity	similar-trend data	none	fixed
Experiment 3	short-term periodicity	own-data	yes	site-specific
Experiment 4	both	similar-trend data	yes	site-specific

* whether training data with a similar-trend to that of the target imputation data or own-data were used.

** whether site-specific features were used to update the APREP-S model.

5.4.1 Experiment 1

We evaluate the model-training and model-operation phases using training data that have a trend similar to that of the target imputation data. We however do not evaluate the update process.

Experimental Settings

Dataset We use a dataset [43] comprising data regarding wireless temperature and humidity sensors (DHT-22) that are installed both inside and outside a home. These sensors, which are widely used, can measure pressure, temperature, humidity, magnetometer, gyroscope, accelerometer, image, etc. In this experiment, we select temperature and humidity as the sensor data because temperature and humidity are numerical and time-series data and are updated on a daily basis.

This dataset has 29 columns, which present data such as measurement time, temperature, humidity, pressure, and wind speed. These data are collected from nine sensors, which are installed on the first floor, second floor, and outside of a house, e.g., sensor 1 measures temperature $T1$ and humidity $RH1$. Four sensors are installed on the first floor; sensor 1 in the kitchen area, sensor 2 in the living area, sensor 3 in the laundry room, and sensor 4 in the office room. Sensors 1 and 2 are located in the same room. Furthermore, five sensors are installed on the second floor, namely, sensor 5 in the bathroom, sensor 6 in the north direction outside the house, sensor 7 in the ironing room, sensor 8 in the children's room, and sensor 9 in the parents' room. These data are collected over 137 days (4.5 months) and there are 19,735 rows of data per sensor. Each sensor transmits data approximately every 3.3 min, which are then aggregated from 3.3 to 10 min. The digital DHT-22 sensors used in the experiment have an accuracy of $\pm$ 0.5°C for temperature measurements and $\pm$ 3% for relative humidity measurements. We generate the experimental data including the outliers and missing data based on this dataset. Let the probability of occurrence of missing data depend

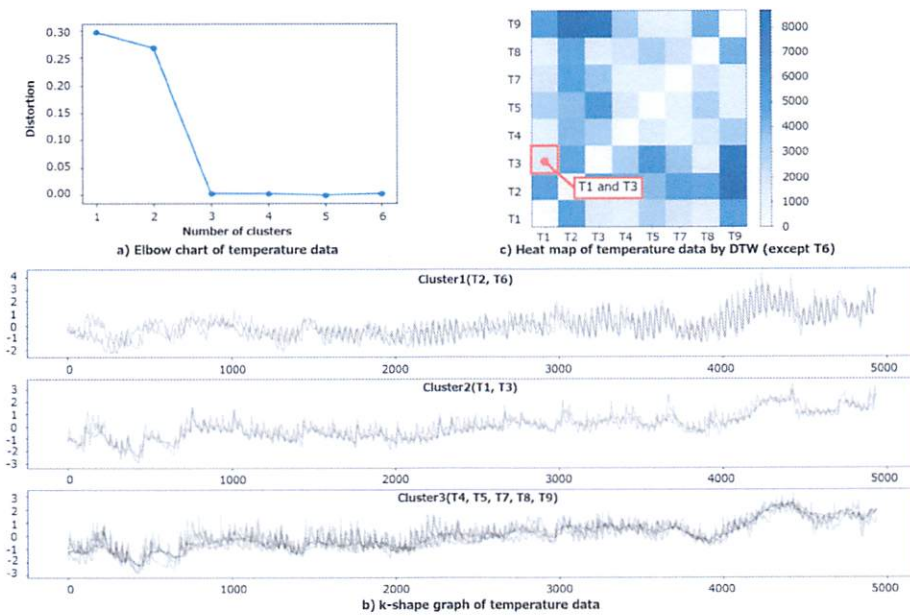

Fig 5.12. Result of similarity of temperature (T) data.: a) elbow chart, b) classification of k-Shape (T data classified three clusters) c) heat map of dynamic time warping (DTW) (deep blue color denotes a large difference, while light blue color denotes a small difference).

on the exponential distribution.

$$f(e) = \frac{1}{\epsilon} \exp(-\frac{e}{\epsilon}) \quad (500 \leq \epsilon \leq 1000). \tag{5.8}$$

We configure a total of 9 outliers and 1 missing data point from every 10 data points. The outlier difference between the original and the experimental data is estimated based on a Gaussian distribution.

$$\mathcal{N}(e; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(e-\mu)^2}{2\sigma^2}\right\} \tag{5.9}$$

where μ is the mean and σ^2 is the variance of the Gaussian distribution.

We calculate the rate of similarity of the experimental data using "k-Shape" for classification, as mentioned in Section 3.3. In this experiment, we extract data every 30 min from the original dataset to calculate similarity. The results are presented in Figs. 5.12 and 5.13. First, we must decide a cluster number to be inputted into the k-Shape using an elbow chart to determine the number of clusters in the chart (a). For elbow charts, the best T's cluster number is 3 and the cluster number of RH is 4. The k-Shape results are indicated by the line graph (b). T is classified into three

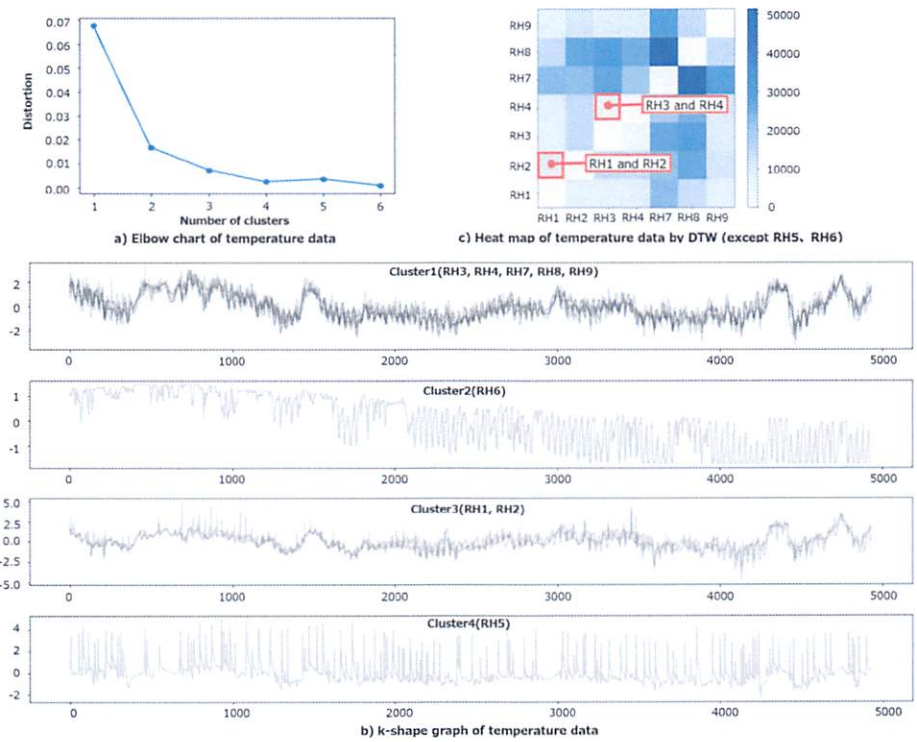

Fig 5.13. Result of similarity of humidity (RH) data.: a) elbow chart, b) classification of k-Shape (RH data are classified into four clusters), c) heat map of DTW (deep blue color denotes a large difference, while light blue color denotes a small difference).

clusters: cluster 1=$[T2, T6]$, cluster 2=$[T1, T3]$, and cluster 3=$[T4, T5, T7, T8, T9]$. RH is classified into four clusters: cluster 1=$[RH3, RH4, RH7, RH8, RH9]$, cluster 2=$[RH6]$, cluster 3=$[RH1, RH2]$, and cluster 4=$[RH5]$. Additionally, we calculate DTW [44] [45], which detects patterns in a data stream or time series based on the distance between the data. The distance of DTW is shown as a heat map (c). The deeper the blue color, the larger the difference between two data. It should noted that T's heat map does not contain $T6$, and RH's heat map does not contain both $RH5$ and $RH6$ because the values are extremely difference than others as the sensor that measures $T6$ and $RH6$ is installed outside the house and the one that measures $RH5$ is installed in the bathroom.

We choose three pairs of data: [training data, target imputation data]=[$T1$, $T3$], [$RH1$, $RH2$], [$RH2$, $RH1$], and [$RH3$, $RH4$]. Each pair is classified in the same cluster by the k-Shape and indicated by a light color in the DTW heat map. $T1$ is configured to have 20 outliers and missing data with Eq. (5.8) and Eq. (5.9), whereas $T3$, $RH1$, $RH2$, $RH3$, and $RH4$ have 39, 36, 37, 20, and 38 outliers, respectively.

Table 5.4. Imputation models $\mathcal{M}$.

m_1	mean of just front and behind
m_2	input front data
m_3	spline interpolation

Imputation Model We define three imputation models, m_1, m_2, and $m_3 \in \mathcal{M}$ for this experiment. m_1 is the mean of the front and rear data, m_2 is an imputation of the front data without the transform, and m_3 is spline interpolation [23]. In this experiment, we define single imputation methods because it is a well-used imputation method. A summary of the imputation models is provided in Table 5.4. The imputation models are

- $m_1(o_d) = (o_{d-1} + o_{d+1})/2$.
- $m_2(o_d) = o_{d-1}$.
- $m_3(o_d) = a_j(o_d - o_j)^3 + b_j(o_d - o_j)^2 + c_j(o_d - o_j) + d_j \quad (1 \leq j \leq D-1)$.

where d $(d = 1, ..., D)$ is a target imputation data.

Feature We define three features as follows..

- Gradient between the target imputation areas: $v_b + v_a/D$, where v_b and v_a lie immediately before and after the values of the target imputation area, and D is the number of rows in the target imputation area.
- Gradient trend before the target imputation area. $(v_{b-1} + v_b)$, where v_b lies immediately before the target imputation area, and v_{b-1} lies immediately before v_b.
- Difference between the mean of the before and after values and the mean of all the data: $|(v_b + v_a)/2| - \text{mean}(v_{all})$, where v_b and v_a lie immediately before and after values of the target imputation area, and v_{all} is the target imputation data.

Experimental Method

In this experiment, the sum of the squares of the errors between the original data and the imputation values of APREP-S in the target imputation area is compared. Let the original data be $\boldsymbol{Org}$=$(org_1, ..., org_D)$,

$$E = \frac{1}{2}\sum_{d=1}^{D}(org_d - v_d)^2 \tag{5.10}$$

where D is the size of all the target imputation areas, org is the original value, and v_d is the value according to APREP-S or the existing imputation models for comparison with APREP-S.

Three imputation methods are used for comparison: 1) mean of the entire data, 2) mean of the around-the-target imputation data, and 3) spline interpolation. The model that corresponds to a smaller E is the one with higher accuracy. We use sensor data every 10 min. The range of the target data is defined for each of the 6 hours before and after the target imputation data; this corresponds to 72 rows. The mean of the entire data method uses the mean of all the target imputation data as its input. The mean of the around-the-target imputation data method uses the mean of the 6 hours of data before and after the target imputation data as its input. This corresponds to 12 hours, with 72 rows. The spline interpolation method uses the median of the list that has 73 rows from the model that learns based on the original 72 rows of data as its input.

Experimental Procedure

In the model-training phase, the two parameters of APREP-S are α and β, and both of them is Gaussian distributioni $\mathcal{N}(0, 2)$. Furthermore, an analyst inputs Y, which is a selected imputation model of the target imputation area. It is shown below for each target imputation model on the target imputation data. $\boldsymbol{Y}$ exists in each target imputation data, e.g., $\boldsymbol{Y}_{T1} = [3, 1, 1, 3, 2, 3, 1, 1, 2, 2, 3, 2, 3, 2, 3, 3, 3, 1, 3, 3]$ (the list size is 20), where $\boldsymbol{Y}_{T1}$ is $\boldsymbol{Y}$ of $T1$. The steps in the generation of the APREP-S model are as follows:

1. Searching outliers and missing data (target imputation data) and generating $\boldsymbol{Y}$.
2. Calculating the features of target imputations and normalization, and generating $\boldsymbol{X}$.
3. Inferring APREP-S model parameters, α and β using Algorithm 2. Inputting pair of $\boldsymbol{X}$ and $\boldsymbol{Y}$. The target imputation models are $\mathcal{M}$. The output is the APREP-S model.

In the mode-operating phase, for the target imputation data, we search the target imputation area and calculate the features and training data. Subsequently, we infer the likelihood of each method for the target imputation area by using the APREP-S model. For example, if the target imputation data is $T3$, the APREP-S model is generated from $T1$, which is a similar-trend pair of $T3$. The recommendation proportions, $\boldsymbol{P}$, of the first three imputation targets are as follows:

Table 5.5. Comparison of accuracy using sum-of-squares error E (Eq. (5.10)).

Training data	Inference data	APREP-S	All	Around	Spline
$T1$	$T3$	5.81	88.96	2.61	0.20
$RH1$	$RH2$	0.15	175.18	16.87	0.99
$RH2$	$RH1$	355.49	935.75	683.26	526.80
$RH3$	$RH4$	0.16	370.97	8.40	0.21

(*) "All" indicates mean of the entire data.
"Around" mean of the around-the-target imputation data.
"Spline" means spline interpolation.

Table 5.6. Feature of experimental data.

	$RH1$	$RH2$	$RH3$	$RH4$	$T1$	$T3$
changing point(*)	7/36	5/37	2/20	5/38	2/20	2/39
percentage	19%	14%	10%	13%	10%	5%

(*) changing point : *changing points / all imputation targets.*

1st: $m_1 = 39.67\%$, $m_2 = 4.97\%$, $m_3 = 55.36\%$.

2nd: $m_1 = 9.02\%$, $m_2 = 0.22\%$, $m_3 = 90.76\%$.

3rd: $m_1 = 12.23\%$, $m_2 = 0.22\%$, $m_3 = 87.55\%$.

Then, APREP-S calculates imputation values of each $\mathcal{M}$. The values $\boldsymbol{v}$ of the first three imputation targets are as follows.

1st: $m_1 = 20.60$, $m_2 = 20.60$, $m_3 = 21.00$.

2nd: $m_1 = 19.34$, $m_2 = 19.29$, $m_3 = 19.70$.

3rd: $m_1 = 20.13$, $m_2 = 20.20$, $m_3 = 19.78$.

In this experiment, we assume that the analyst selects the method that has the highest probability. As a result of inference $T3$ by APREP-S, the selected method number list of $T3$ is $\boldsymbol{Y}_{T3}$=[3, 3, 3, 1, 1, 2, 3, ..., 3, 3] (the list size is 39).

Experimental Result and Discussion

The accuracy results are presented in Table 5.5. The sum-of-squares error, E (Eq. (5.10)), is calculated using the following methods: APREP-S, mean of the entire data, mean of the around-the-target imputation data, spline interpolate, and original data. For deriving the inference of $RH2$ based on $RH1$, $RH1$ based on $RH2$, and $RH4$ based on $RH3$, APREP-S yields the most accurate value. For deriving the inference of the $T1$ and $T3$ pairs, the spline method yields the most accurate values, whereas APREP-S

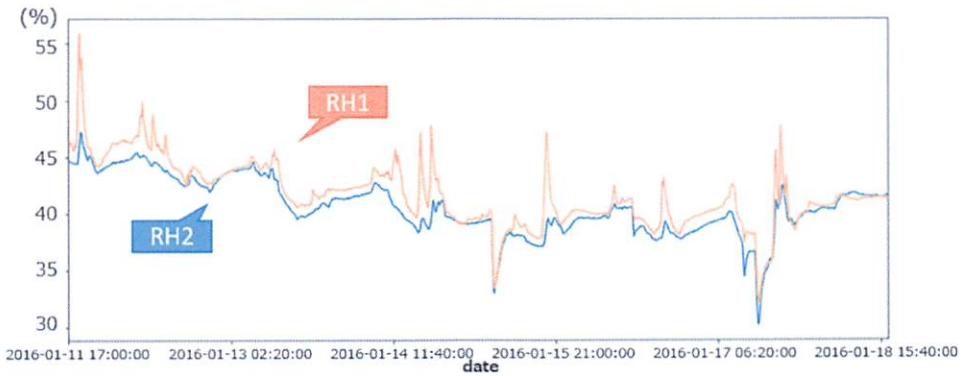

Fig 5.14. Line graph of T and RH data during a week.

yields the third highest accurate result. Single imputation shows the worst accuracy in all the pairs of training and inference data. Therefore, APREP-S is the most suitable method for RH data, although not the best method for T data.

We conclude that APREP-S is more suitable for fluctuating data with several data points whose values are constantly changing. The number of such data points and the percentage are presented in Table 5.6. As can be seen, there are more target imputation data at the changing point in RH data than in T data, for example, $T3$ has only two changing points in the target imputation data, whereas $RH1$ has seven. In this experiment, we create the imputation values randomly. RH data fluctuates more than T data; therefore, the possibility that the changing points become the target imputation data is higher than that of T data. The line graphs of RH and T data for one week are depicted in Fig. 5.14. The above lines indicate the RH data, while the below lines indicate the T data.

Although the accuracies of spline interpolation and mean are sufficient for gentle data, they are not optimal as the imputation of the changing points. On the contrary, the accuracy of APREP-S is not too low for gentle data, but is the highest for the imputation of the changing points. Therefore, we conclude that APREP-S is suitable for fluctuating data such as humidity data, human motion data, and trajectory data.

The summary of the results of this experiment is as follows.

- To generate the APREP-S model, we can use data having trends similar to that of the target imputation data as training data.
- Comparison of APREP-S with well-known imputation methods revealed that selecting imputation methods by the target imputation area is efficient.
- From the experimental result, we conclude that APREP-S tends to be more

suitable for fluctuating data. It is sufficiently accurate even for imputation by only mean and only spline interpolation in the gentle data.

5.4.2 Experiment 2

Experiment 2 is an extension of Experiment 1 in the types of the imputation methods used in APREP-S and the range of the target imputation area. In Experiment 2, we compare the APREP-S method with the generalized additive model (GAM) and RNN methods. These methods are defined in APREP-S as candidate methods. We use data having a similar-trend to that of the target imputation data as training data. Similar to Experiment 1, this we do not evaluate the update process in Experiment 2.

Experimental Settings

Dataset We used two datasets in this experiment. One dataset comprises the temperature and humidity estimated by a temperature and humidity sensor, and the other comprises the weather data of the local area estimated by another temperature and humidity sensor.

The dataset comprising temperature and humidity data comprises data estimated by wireless sensors (DHT-22) installed both inside and outside a home [43]. This dataset has 29 columns, which present data such as measurement time, temperature, humidity, pressure, and wind speed. The temperature and humidity data are obtained from nine sensors, which are installed on the first floor, second floor, and outside a house. These data are collected over 137 days (4.5 months), and there are 19,735 rows of data per sensor. Each sensor transmits data approximately once every 3.3 min, and subsequently, the data are aggregated from 3.3 to 10 min. The digital DHT-22 sensors used in the experiment have an accuracy of ± 0.5°C for temperature measurements and ± 3% for relative humidity measurements. Although the original data has 29 columns, we extract the data from only two humidity columns for this experiment, the kitchen area ($RH1$) and the living room ($RH2$). These two columns are selected because they both contain data estimated by sensors that are installed in the same room on the first floor and have similar data trends. Fig. 5.15 depicts the line graphs of $RH1$ and $RH2$, which present the data collected from the first week of the experiment.

The training and experimental data are generated based on $RH2$ and $RH1$, respectively. We then configure the missing values in these data. Let the probability of occurrence of missing data and the number of continuous missing data depend on a Gaussian distribution.

$$\mathcal{N}(e;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(e-\mu)^2}{2\sigma^2}\right\} \tag{5.11}$$

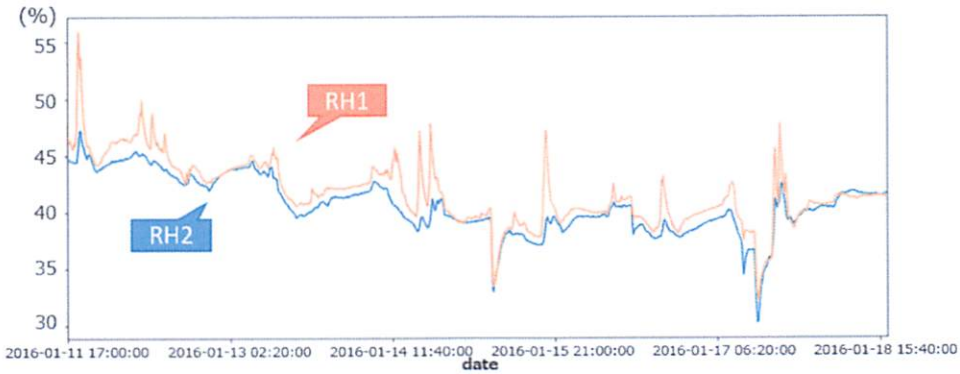

Fig 5.15. Line graph of $RH1$ and $RH2$ in a week from 2016-Jan-11 to 2016-Jan-18.: orange denotes $RH1$ data, blue $RH2$ data. ©2019 IEEE in literature [8]

where μ and σ^2 are the mean and variance of the Gaussian distribution, respectively. The probability of missing data is $\mathcal{N}(0, 144)$ (because each day has 144 corresponding data rows), and the number of continuous missing data is $\mathcal{N}(1000, 3000)$. We generate a total of 928 rows of missing data for $RH1$, subdivided into 4 target imputation areas and 733 rows of missing data for $RH2$, subdivided into 4 target imputation areas.

The data in the weather dataset are measured in the local area where the sensor is located, namely, Stambruges, which is located at a distance of 24 km from the city of Mons in Belgium [46]. This dataset includes data of temperature, wind speed, humidity, and pressure from July 1, 2008 to June 20, 2019, recorded once every hour. Since the data is recorded once every hour, for example, the entry corresponding to the period 1:00:00 to 1:59:59 is inputted the data point at 1:00:00 in this experiment.

Imputation Model In this experiment, we define four imputation methods for APREP-S: m_1, m_2, m_3, and $m_4 \in \mathcal{M}$. m_1 is the mean of the lower and upper limits of the target imputation interval, m_2 is Fbprophet, a well-known (GAM) [27] [47], m_3 is LSTM, and m_4 is spline interpolation. A summary of the imputation models is presented in Table 5.7.

The mean, m_1, is

$$f(v) = \frac{1}{2}(v_b + v_a) \tag{5.12}$$

where v_b and v_a are the immediately before and immediately after values of the target imputation area.

Table 5.7. Imputation models $\mathcal{M}$.

m_1	mean
m_2	Fbprophet
m_3	LSTM
m_4	spline interpolation

The Fbprophet, m_2, is based on GAM, which is described in Section 2.2.2, and

$$g(t) = \frac{C(t)}{1 + \exp(-(k + \boldsymbol{a}(t)^{\mathrm{T}}\boldsymbol{\delta})(t - (m + \boldsymbol{a}(t)^{\mathrm{T}}\boldsymbol{\gamma})))}. \tag{5.13}$$

Now,

$$\gamma_j = \left(s_j - m - \sum_{i<j} \gamma_l\right)\left(1 - \frac{k + \sum_{i<j} \delta_l}{k + \sum_{i \leq j} \delta_l}\right) \tag{5.14}$$

$$a_j(t) = \begin{cases} 1, & \text{if } \mathrm{t} \leq \mathrm{s_j} \\ 0, & otherwise \end{cases} \tag{5.15}$$

where $C(t)$ is a time-varying capacity, s_j is the changing points, k is the base rate at time t, and $\delta \in \mathbb{R}^s$ is a vector of rate adjustments.

$$s(t) = \sum_{n-1}^{N} \left(a_n \cos\left(\frac{2\pi nl}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right)\right) \tag{5.16}$$

where a and b are parameters, and P is the regular period that we expect the time-series to have, e.g., $P = 365.25$ for yearly data or $P = 7$ for weekly data.

$$h(t) = Z(t)\boldsymbol{\kappa} \tag{5.17}$$

$$Z(t) = [\mathbf{1}(t \in D_1), ..., \mathbf{1}(t \in D_L)]. \tag{5.18}$$

LSTM, m_3, uses Keras of TensorFlow. To generate a model, LSTM extracts training data from every third row of the dataset. The step and batch sizes are both 144, which is also the number of data points collected in 3 days. The number of hidden units and the number of iterations of training are 100 and 200, respectively.

Spline interpolation, m_4, is described in Section 2.2.1.

Feature The following two types of features are defined: the features calculated from the target imputation data and the ones calculated from extraneous data. By defining features for data other than its own, APREP-S can recommend pairs of proportion and value in a single target imputation area. In this experiment, four features are calculated from the target imputation data, and three weather data are calculated from the extraneous data.

1. Number of imputation target rows.
2. Time zone of imputation targets.: "0" if the time zone is 0:00-5:59, "1" if the time zone is 6:00-11:59, "2" if the time zone is 12:00-17:59, and "3" if the time zone is 18:00-23:59.
3. Gradient between target imputation area.: $(v_b - v_a)/D$.
4. Trend of the gradient of the lower and upper limits.: comparing the trend of the preceding area with that of the succeeding area. If both the trends are positive, input "1." If the two trends are opposite, input "-1."
5. Temperature (°C) from the weather data.
6. Type of the weather from the weather data.
7. Humidity from the weather data.

Experimental Method

We compare the accuracy and the similar trend of APREP-S with those of existing methods as follows: 1) the accuracy by calculating the sum of squares of errors, and 2) the similarity by calculating the k-Shape, which is described in 3.3. The imputation values for the APREP-S are calculated in the model-operating phase.

1. Accuracy by sum-of-squares error E: the model with higher accuracy is the one with smaller E.

$$E = \frac{1}{2}\sum_{d=1}^{D}(org_d - v_d)^2 \tag{5.19}$$

where D is the size of all the target imputation areas, org is the original value, and v_d is the value according to APREP-S or the existing imputation models for comparison with APREP-S. The model that corresponds to a smaller E has a higher accuracy.

2. Similarity by k-Shape: k-Shape is a clustering method for time-series data that focuses on amplitude-scaling invariance and time-shifting invariance. The data contained in the same cluster as the original data are the most similar data.

We compare four existing models, which are the same as those defined in the APREP-S: mean, GAM (Fbprophet), LSTM, and spline interpolation. Similar to the case of APREP-S, the periodicity of Fbprophet is configured as daily, and the step size of LSTM method as 144, which is same as the batch size and the total number of data in 3 days. The number of hidden units and the number of training iterations are 100 and 200, respectively.

Experimental Procedure

If the period of the target imputation area is less than half-day and the trend of the gradient is positive, the spline interpolation method m_4 is chosen, whereas, when the gradient is negative, the mean method m_1 is chosen. If the period of the target imputation area is greater than half-day and the gradient is positive, LSTM method m_3 is chosen, whereas when the gradient is negative, Fbprophet method m_2 is chosen. From this training data, APREP-S infers the parameters, α and β, and generates the APREP-S model. Subsequently, the analyst selects the method with the highest proportion as the most suitable method.

Experimental Result and Discussion

The results of the comparison of APREP-S and the existing methods are presented in Table 5.8. APREP-S shows the smallest value, followed by mean, and then by Fbprophet. Although spline interpolation is one of the most accurate methods when the number of imputation areas is one, it provides the worst accuracy in this experiment. The accuracy of APREP-S is 1.4 times that of mean, 1.6 times that of Fbprophet, 2.3 times that of LSTM, and 2.4 times that of spline interpolation.

Owing to the similarity of their imputation values, APREP-S is deemed to be the best approximation for the original data. The associated line graphs of all the methods are depicted in Fig. 5.16, and the result of k-Shape is shown in Fig. 5.17. $RH1$ has four imputation areas; therefore, these graphs can be divided into four areas. Fig. 5.16 presents the values of the original data and those calculated by each imputation method. In this experiment, the number of clusters is four, and cluster 1 = [original data, APREP-S], cluster 2 = [spline], cluster 3 = [Fbprophet], and cluster 4 = [mean, LSTM] in Fig. 5.17. In other words, the trend of APREP-S is the most similar to that of the original data. The result of the mean method, which is indicated by a blue line, is a straight line for each period. Therefore, it does not express the

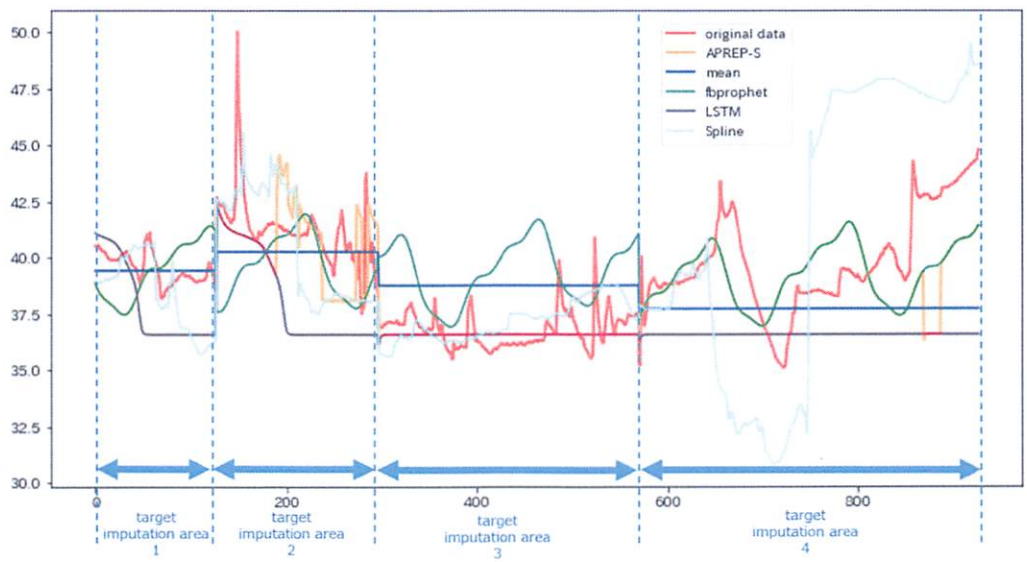

Fig 5.16. Line graph of original data, APREP-S, and Existed Imputation Methods in All Imputation Area.: Red denotes original data, orange APREP-S, blue mean, green Fbprophet, purple LSTM, light blue spline interpolation. The inference data has four target imputation area. ©2019 IEEE in literature [8]

Table 5.8. Result of sum-of-squares error E (Eq. (5.19)).

APREP-S	Mean	Fbprophet	LSTM	Spline
1802.6	2580.4	2843.2	4177.0	6949.0

time-series trend. The result of Fbprophet method, which is indicated by the green line, shows the same trend and periodicity in all target imputation areas. However, the trends and periodicity of the original data and the result of Fbprophet do not match.

We propose sampling the training data in the model-training phase. In this experiment, we did not focus on the sampling method. However, the sampling methods must be discussed depending on the goal of the analysis and its criteria. In this proposal, similarity of the training data is essential for inference. The result of the sum-of-squares error for the different sampling methods is presented in Table 5.9 regarding 1) the extraction of the first 1.5 months of data from the training data and 2) sampling every 30 min of the training data. In the LSTM method, the data at every 30 min is 2.3 times that of the first 1.5 months. The accuracy of APREP-S at every 30 min is worse than first 1.5 months as well. The similarity was however the same as that of the training data of the first 1.5 months; cluster 1 = [original data, APREP-S], cluster 2 = [spline], cluster 3 = [Fbprophet], and cluster 4 = [mean, LSTM]. We consider this

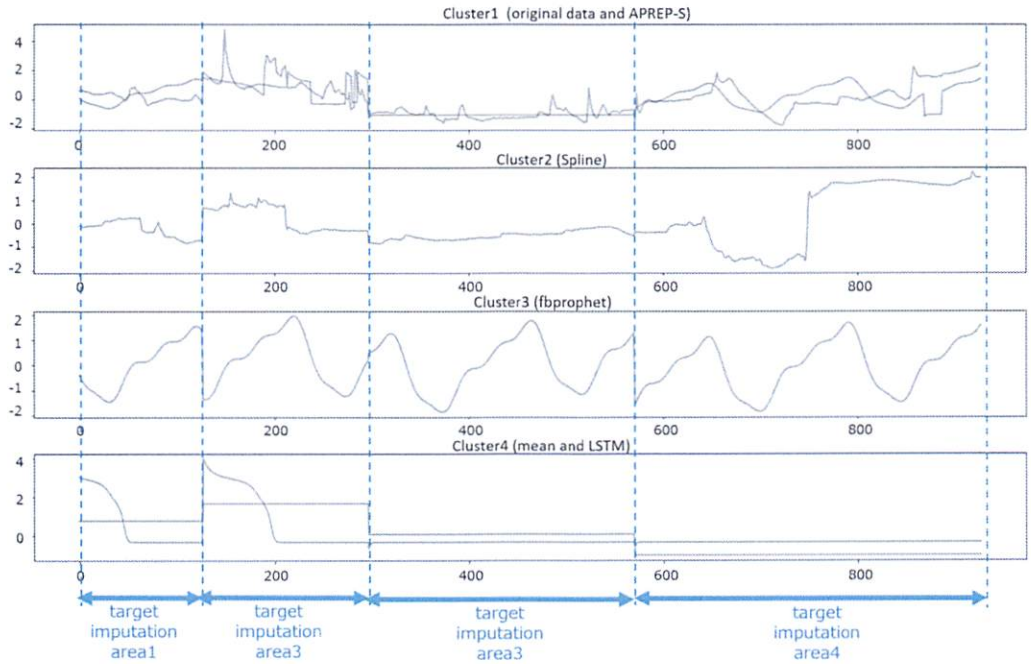

Fig 5.17. Similarity between each imputation models. A cluster indicates a similarity group of the time-series data trend.: cluster 1 has original data and APREP-S, cluster 2 has spline interpolation, cluster 3 has Fbprophet, and cluster 4 has mean and LSTM. ©2019 IEEE in literature [8]

Table 5.9. Comparing E (Eq. (5.19)) on the sampling method of LSTM model.

Method of sampling	APREP-S	LSTM
first 1.5 months	1802.6	4177.0
every 30 min	3338.3	9589.7

reason that the inference data is every 10 min data. In the experiment, some of the training data were lost during sampling. Therefore, APREP-S can infer the time-series trend; however, its accuracy is slightly lower than that for the first 1.5 months of data, which is sampled every 10 min.

The summary of the results of this experiment is as follows.

- Found that APREP-S is a high accuracy model in the target imputation area with a size more than one. APREP-S uses some imputation methods including those of machine learning.
- Verified APREP-S regarding two aspects: accuracy and similar-trend of time-series data. It was revealed that APREP-S is one of the most effective imputation methods for time-series data.

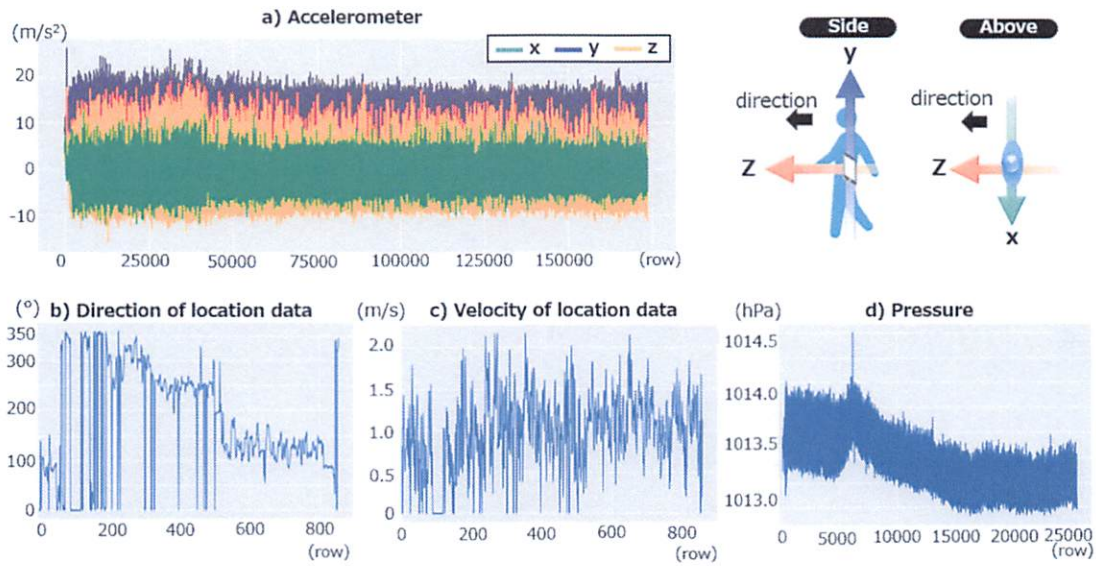

Fig 5.18. Walking data measured by phyphox: 25,000 rows of accelerometer values were collected in approximately 1 min, 200 rows of GPS data were obtained in approximately 3.3 min, and 5,000 rows of pressure data were recorded in approximately 2.8 min.

5.4.3 Experiment 3

We evaluated the update process with site-specific features using short-term periodicity data. We use the same data as the training and inference of APREP-S. It is termed as the own-data of the target imputation data. We evaluated APREP-S to compare 1) the model updated using the site-specific features with the model updated using all features defined in the model training phase and 2) the performance of APREP-S, which generated new learning data using HMM in the model updating phase with that of existing imputation methods. For trajectory analysis and behavioral analysis in a case of a factory, we used human activity data.

Experimental Settings

Dataset We measured the accelerometer (without g), GPS, and pressure data for two activities using a smartphone: 1) walking around our university campus, and 2) ascending 120 stairs from the first floor to the 6th floor of the university building; the sensor was placed on the left side of the waist. In this experiment, we used the accelerometer data as inference data, and the GPS data and pressure data as features. The walking data were collected during a period of approximately 14.5 min, and it generated 177,884 rows of accelerometer data, 872 rows of GPS data, and 26,281 rows

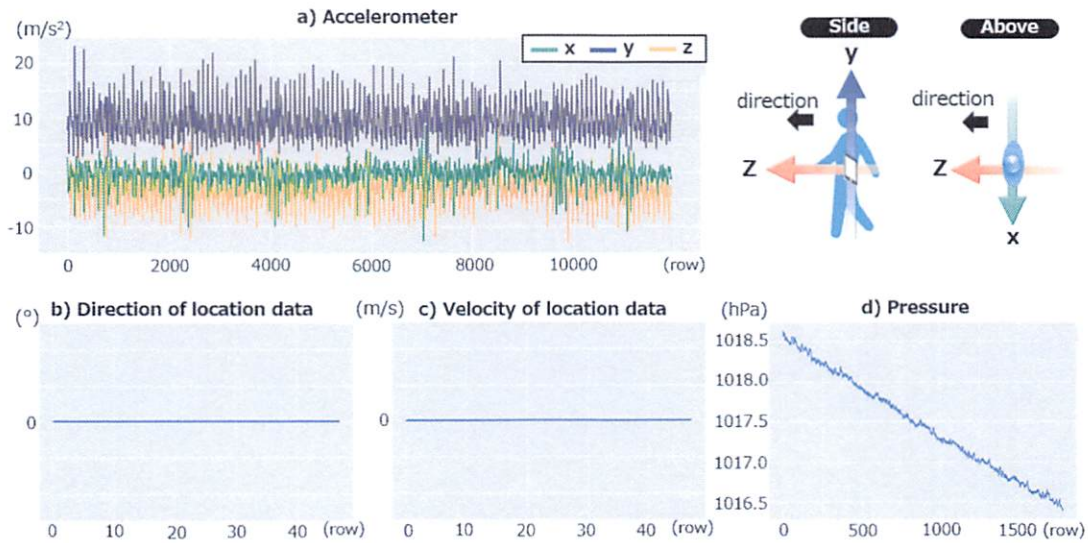

Fig 5.19. Ascending stairs data measured by phyphox: 2,000 rows of accelerometer collected in approximately 10 s, 10 rows of GPS data, and 500 rows of pressure data measured in approximately 15 s.

of pressure data. When ascending the stairs, data were recorded for approximately 1.5 min, and they generated 16,132 rows of accelerometer data, 64 rows of GPS data, and 2,382 rows of pressure data. We discarded the first and last 10 s of data, respectively, to set the sensor on the waist. The recorded data are shown in Fig. 5.18 and Fig. 5.19. We used the x- and y-axes of the accelerometer data as the target imputation data. As the features of the accelerometer data, we used the velocity and direction to update the model of the walking data, and we used the pressure data to update the model of the data collected when ascending the stairs. We configured the missing data on the x- and y-axes of the accelerometer data. Let the probability of the occurrence of missing data and the number of continuous missing data points depend on the Gaussian distribution.

$$\mathcal{N}(e;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(e-\mu)^2}{2\sigma^2}\right\} \tag{5.20}$$

where μ and σ^2 denote the mean and variance of the Gaussian distribution, respectively. The probability of missing a consecutive number is $\mathcal{N}(0, 200)$ for every $\mathcal{N}(0, 2000)$ times. Thus, the accelerometer x- and y- axis data for walking generated 1,582 and 1,374 rows of the target imputation area, respectively. The accelerometer x- and y-axes data for ascending the stairs generated 385 and 138 rows of the target imputation area

We measured the sensor data using phyphox [48]. Phyphox is a smartphone ap-

Table 5.10. Units of sensor data.

Name	Unit
accelerometer	m/s^2
height	m
velocity	m/s
direction	°
horizontal accuracy	m
vertical accuracy	m
pressure	hPa

Table 5.11. Imputation models $\mathcal{M}$.

m_1	mean
m_2	Fbprophet
m_3	GRU
m_4	spline interpolation

plication that records data using the built-in sensors of the smartphone such as the accelerometer, magnetometer, gyroscope, light sensor, pressure meter, proximity sensor, microphone, and GPS.

In phyphox, the x-axis points to the right side of the screen while looking at the screen in portrait orientation. The y-axis is upward along the long side of the screen. The z-axis is perpendicular to the screen and positive in the direction of the screen. The accelerometer data comprise three axes, x, y, and z, with units of m/s^2. The GPS data from the satellite are composed of the latitude, longitude, height [m], velocity [m/s], direction [°], horizontal accuracy [m], and vertical accuracy [m]. The pressure data is composed only of pressure [hPa] and is designed to determine the vertical position of the user within a building, which is approximately 0.1 hPa = 0.1 mbar. These units are summarized in Table 5.10.

Imputation model APREP-S includes four imputation methods, $m_1, m_2, m_3, m_4 \in \mathcal{M}$, as listed in Table 5.11. m_1 represents the mean value of the addition and division by two values for each value before and after the target imputation area,

$$f(v) = \frac{1}{2}(v_b + v_a) \tag{5.21}$$

where v_b and v_a indicate the values before and after the target imputation area, respectively. m_2 denotes Fbprophet, which is based on GAM described in Section 2.2.2, and

$$g(t) = \frac{C(t)}{1 + \exp(-(k + \boldsymbol{a}(t)^{\mathrm{T}}\boldsymbol{\delta})(t - (m + \boldsymbol{a}(t)^{\mathrm{T}}\boldsymbol{\gamma})))}. \tag{5.22}$$

Now,

$$\gamma_j = \left(s_j - m - \sum_{i<j} \gamma_l\right)\left(1 - \frac{k + \sum_{i<j} \delta_l}{k + \sum_{i \leq j} \delta_l}\right) \tag{5.23}$$

$$a_j(t) = \begin{cases} 1, & \text{if } t \leq s_j \\ 0, & otherwise \end{cases} \tag{5.24}$$

Let $C(t)$, s_j, k, and $\delta \in \mathbb{R}^s$ denote a time-varying capacity, changing points, base rate at time t, and vector of rate adjustments, respectively.

$$s(t) = \sum_{n-1}^{N} \left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right) \tag{5.25}$$

where a and b denote parameters, and let P denote the regular period that we expect the time-series to have, for example, $P = 365.25$ for yearly data or $P = 7$ for weekly data.

$$h(t) = Z(t)\kappa \tag{5.26}$$

$$Z(t) = [\mathbf{1}(t \in D_1), ..., \mathbf{1}(t \in D_L)]. \tag{5.27}$$

m_3 is a gated recurrent unit (GRU) [49] of one of the RNN architectures. This method learns to encode a variable-length sequence into a fixed-length vector representation and to decode a given fixed-length vector representation back into a variable-length sequence. m_4 represents spline interpolation.

Feature Six features were used in this experiment. Features 1-6 are used for the initial model in the model training phase, features 4 and 5 are used for updating the walking data model as specific features of walking data, and feature 6 is used for updating the ascending stairs model as specific features of ascending stairs. The features used for each activity are listed in Table 5.12.

1. Continuous number of rows in the target imputation area.
2. Gradient between target imputation areas,

   $$(v_b - v_a)/D \tag{5.28}$$

   where v_b and v_a denote the values of the target imputation area just before and just after, and D represents the number of rows in the target imputation area.
3. Gradient trend before and after the target imputation area by comparing the before area trend with the after area trend. Input "1" if both trends are positive, input "-1" if the trends are opposite.
4. Velocity of GPS data.
5. Direction of GPS data.
6. Pressure data.

Table 5.12. Features using each activity for the APREP-S model.

walking	ascending stairs
1, 2, 3, 4, 5	1, 2, 3, 6

* the numbers represent the feature number in Section 5.4.3.

Experimental Method

We evaluated APREP-S by assessing the accuracy and similarity of the trend in the imputation values. We first compared the accuracy of APREP-S with that of existing methods, and then, we compared the similarity of the trend of the original data with the existing method, in which the accuracy is higher than that of APREP-S. Their details are described as follows.

1. Accuracy: calculating the sum-of-squares error (E) of the original data and the results of APREP-S and existing methods. A smaller E indicates higher accuracy.

$$E = \frac{1}{2}\sum_{d=1}^{D}(org_d - v_d)^2 \tag{5.29}$$

   where D denotes the number of target imputation rows, org is the original value, and v_d is the value according to APREP-S or the existing methods it is being compared to.

2. Similarity: comparing the trend of APREP-S with the original data by using the k-Shape. The data distributed in the same cluster as the original data exhibit a similar trend. Details of the k-shape are provided in Section 3.3.

In this experiment, we compare APREP-S with the four existing methods: 1) the mean value of the before and after in the target imputation area, 2) Fbprophet as a representative GAM, 3) GRU as a representative RNN, and 4) spline interpolation as a representative single imputation.

Experimental Procedure

In the model training phase, we first generate the initial training data using all features in Section 5.4.3. We define m_1 (mean) if the number of continuous rows is equal to 1 , m_3 (spline interpolation) if the number of continuous rows is less than 100, and the other is m_2 (Fbprophet). APREP-S generates the model from these training data. APREP-S infers the optimal imputation method on each target imputation area, and APREP-S calculates the imputation values by each imputation method. Fbprophet (m_2) is used

to train the model using 3,000 rows of data beforehand. Here, it is configured such that the periodicity is a second, and the Fourier order of the x- and y-axes is 800 and 1,000 for the walking data, respectively. For the ascending stairs data, the periodicity is configured to be a second, and the Fourier order of the x- and y-axes is 600 and 600, respectively. GRU (m_3) is TensorFlow's KerasAPI. GRU (m_3) is TensorFlow's KerasAPI. To generate a model, the training data was extracted from 10,000 rows of data. The step size was 50, and the batch size was 25. The number of hidden units was 200, and the number of training iterations was 100. Spline interpolation (m_4) uses an interpolate library of the SciPy API. Moreover, in the training algorithm of the APREP-S model, the two parameters α and β depend on the Gaussian distribution $\mathcal{N}(0, 2)$. We used the PyMC3 library and configured NUTS as a step method with 4,000 steps.

In the model-updating phase, APREP-S performs the clustering using HMM. HMM is a Gaussian HMM of the hmmlearn library. The number of clusters is eight, which is twice the number of methods, the covariance type is full, and the maximum number of iterations is 300.

Experimental Result and Discussion

The results of the accuracy of the imputation values by the sum-of-squares error are presented for walking data and ascending stairs in Table 5.13. In the feature column, the numbers are the feature number, and "all" means all features are used. The number refers to site-specific features, as described in Table 5.12. In the model column, "initial" means the result of the initial model of the model training phase, and "update" refers to the result of the updated model of the model updating phase.

The results for the site-specific features and "all" were the same, except for the y-axis of walking. This is attributed to HMM returning to the same cluster group. Moreover, data collected for a short period such as ascending stairs, were distributed in only four clusters of x-axis data and two clusters of y-axis data in spite of the specified number of clusters being eight. For the y-axis of walking, the accuracy of the result of the activity-specific features is higher than "all." Furthermore, the results for the y-axis of walking and the x-axis of ascending stairs data are less accurate than the mean. We evaluated the similarity of the trend by the k-Shape for the y-axis of walking and the x-axis of ascending stairs. In addition, because the accuracy of the spline is superior to that of APREP-S for the x-axis of ascending stairs, the spline was also added. The results are shown in Fig. 5.20. For the y-axis of the walking data, cluster 1 = [APREP-S, original], and cluster 2 = [mean]. For the x-axis of ascending stairs, cluster 1 = [mean, spline], and cluster 2 = [APREP-S, original]. The result of APREP-S is distributed in the same cluster as the original data. The similarity of the

Table 5.13. Results of E by sum-of-squares error (Eq. (5.29)).

	Data	Model	Feature	APREP-S	Mean	Fbprophet	GRU	Spline
walking data	x-axis 1,582 rows	initial	1,2,3, 4,5	11,478 (7.26)	14,579 (9.22)	9.447 (5.97)	18,992 (12.01)	13,364 (8.45)
		update	1,2,3 4,5	9,384 (5.93)				
			all	9.384 (5.93)				
	y -axis 1,374 rows	initial	1,2,3 4,5	22,902 (16.67)	17,193 (12.51)	21,570 (15.70)	45,757 (33.33)	33,521 (24.40)
		update	1,2,3 4,5	20.329 (14.80)				
			all	23,333 (33.96)				
ascending data	x-axis 385 rows	initial	1,2,3 6	957 (2.49)	415 (1.08)	957 (2.49)	2,613 (6.79)	747 (1.94)
		update	1,2,3 6	918 (2.38)				
			all	918 (2.38)				
	y-axis 138 rows	initial	1,2,3 6	535 (3.88)	801 (5.80)	535 (3.88)	1,308 (9.48)	1,230 (8.91)
		update	1,2,3, 6	535 (3.88)				
			all	535 (3.88)				

* the value in parentheses is the mean of the sum-of-squares error per row.
* refer to Table 5.12 for the number in the feature column.

trend in comparison with the original data is worse than that of APREP-S.

In terms of the overall accuracy and similarity, we consider APREP-S to be the most optimal method when the site at which the model is updated differs from that at which it is generated. The results show that the accuracy of APREP-S can be maintained by using clustering methods in the model updating phase.

In this experiment, APREP-S is more flexible as a result of clarifying the process of defining the features. Further, we define the update process of APREP-S when data with a similar-trend to that of the inference data do not exist.

The summary of the results of this experiment is as follows.

- Evaluating that APREP-S trains the model by the own-data of the target imputation data. This means that APREP-S can use the analysis without similar-trend data of the target imputation data.

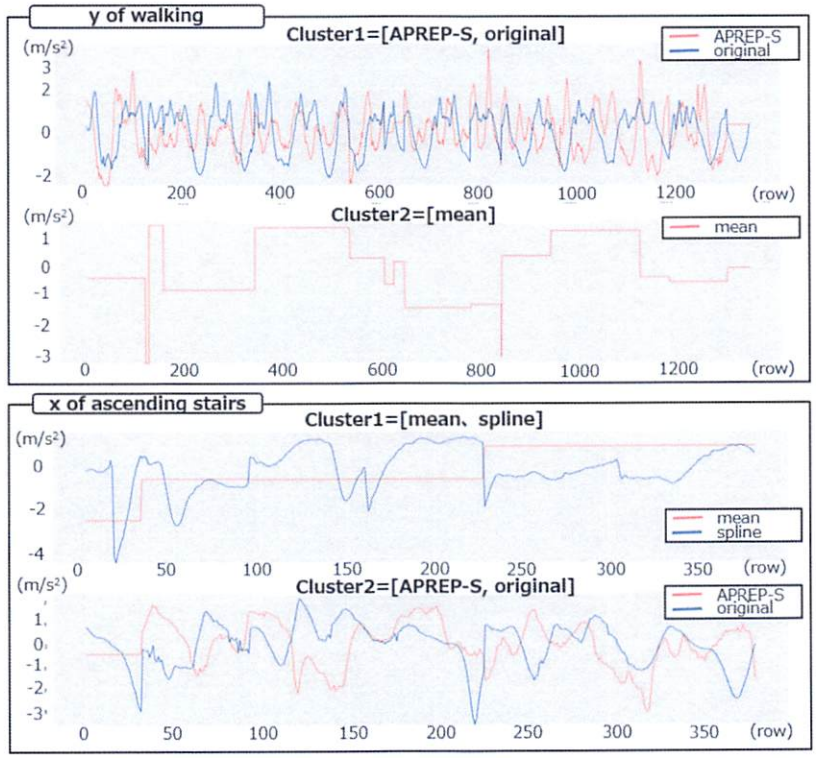

Fig 5.20. Result of k-Shape for y-axis of walking and x-axis of ascending stairs data. Both are extracted only by the target imputation area using site-specific features.

- Finding that APREP-S is an efficient method if the site at which the model is generated differs from that at which the model is updated, because it maintains the accuracy when the site-specific features are selected in the model updating phase.
- A clustering method was defined using the feature data of the inference of the imputation data when APREP-S updates the model. We concluded that APREP-S returned the imputation value with superior accuracy and similarity trend compared with other existing imputation methods.

5.4.4 Experiment 4

We evaluate the update process with site-specific features using short- and long-term periodicity data. We use the similar-trend data to the target imputation data as the training data for APREP-S model.

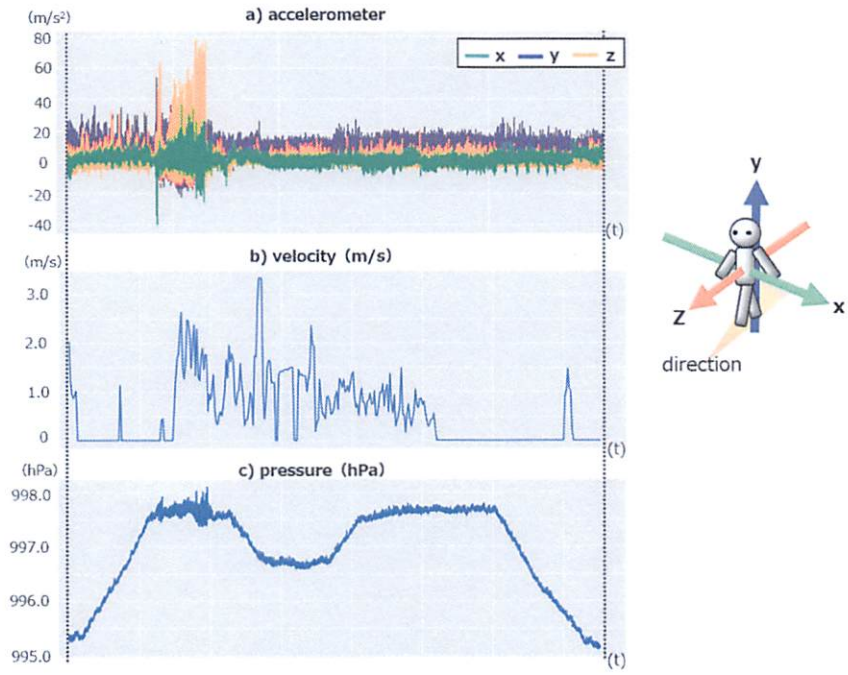

Fig 5.21. Line graph of the combining action data. ©2020 IEEE in literature [9]

Experimental Settings

Dataset Two types of data were used for the experiment: 1) human activity data and 2) temperature and humidity data.

Human Activity Data We used the accelerometer (without g) of a smartphone to measure the location and pressure for four activities: 1) running, 2) walking, 3) ascending stairs, and 4) descending stairs. We combined the data acquired while running, walking, ascending stairs, and descending stairs as the target imputation data, and we used only running, only walking, only ascending stairs, and only descending stairs as the training data for the imputation models in APREP-S. The combined data acquired by the accelerometer, and the velocity and pressure are shown in Fig. 5.21; the duration of each type of activity is listed in Table 5.14.

The data measured by the accelerometer was acquired along three axes x, y, and z in units of $\mathrm{m/s}^2$. The x-axis points to the right side of the screen while looking at the screen in portrait orientation. The y-axis is oriented vertically in the plane of the screen. The z-axis is perpendicular to the screen, and the direction of the screen is positive. The data collected by GPS were received from a satellite and include the

Table 5.14. Duration of sensor and activity.

Time	Activity
0s - 95s	descending stairs
95s - 157s	running
157s - 196s	ascending stairs
196s - 253s	walking
253s - 280s	descending stairs
280s - 410s	walking
410s - 511s	ascending stairs

Table 5.15. Units of sensor data.

Name	Unit
accelerometer	m/s^2
height	m
velocity	m/s
direction	°
horizontal accuracy	m
vertical accuracy	m
pressure	hPa

latitude, longitude, height [m], velocity [m/s], direction[°], horizontal accuracy [m], and vertical accuracy [m]. The pressure sensor data consist only of the pressure [hPa] designed to determine the vertical position of the user in the building [hPa], with hPa = approximately 0.1 mbar. These units are summarized in Table 5.15.

The sensor was positioned at the waist, and the first and last 10 s were excluded as the preparation time for the sensor. The target imputation data were collected during a period of approximately 8.5 min, and the accelerometer data occupied 211,554 rows; the position data, 393 rows; and the pressure data, 15,626 rows. For the training data, running data were acquired during a single period of approximately 10 min, and they comprised 208,330 rows of the accelerometer data. Walking data were only collected for approximately 14.5 min using 355,768 rows of accelerometer data. Data acquired when stairs were ascended were collected for only approximately 1.5 min, and they occupied 32,265 rows of accelerometer data. Data acquired when stairs were descended were collected for approximately 1.1 min and they occupied 28,058 rows of accelerometer data. In addition to the target imputation data, we removed 10 s of data from the beginning and the end of the entire period of activity to set the sensor on the waist. The data is shown in Fig. 5.22.

In this experiment, we configured the missing accelerometer data along the x, y, and z axes in the set of the target imputation data. The probability of the occurrence of missing values is an exponential distribution

$$e(t) = \begin{cases} \lambda e^{-\lambda x} & (x \geq 0) \\ 0 & (x < 0), \end{cases} \tag{5.30}$$

where λ denotes the inverse of the rate parameter and $\lambda = 1/1000$ in this experiment. The number of continuous missing data depends on the Gaussian distribution

$$\mathcal{N}(e; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{(e-\mu)^2}{2\sigma^2} \right\} \tag{5.31}$$

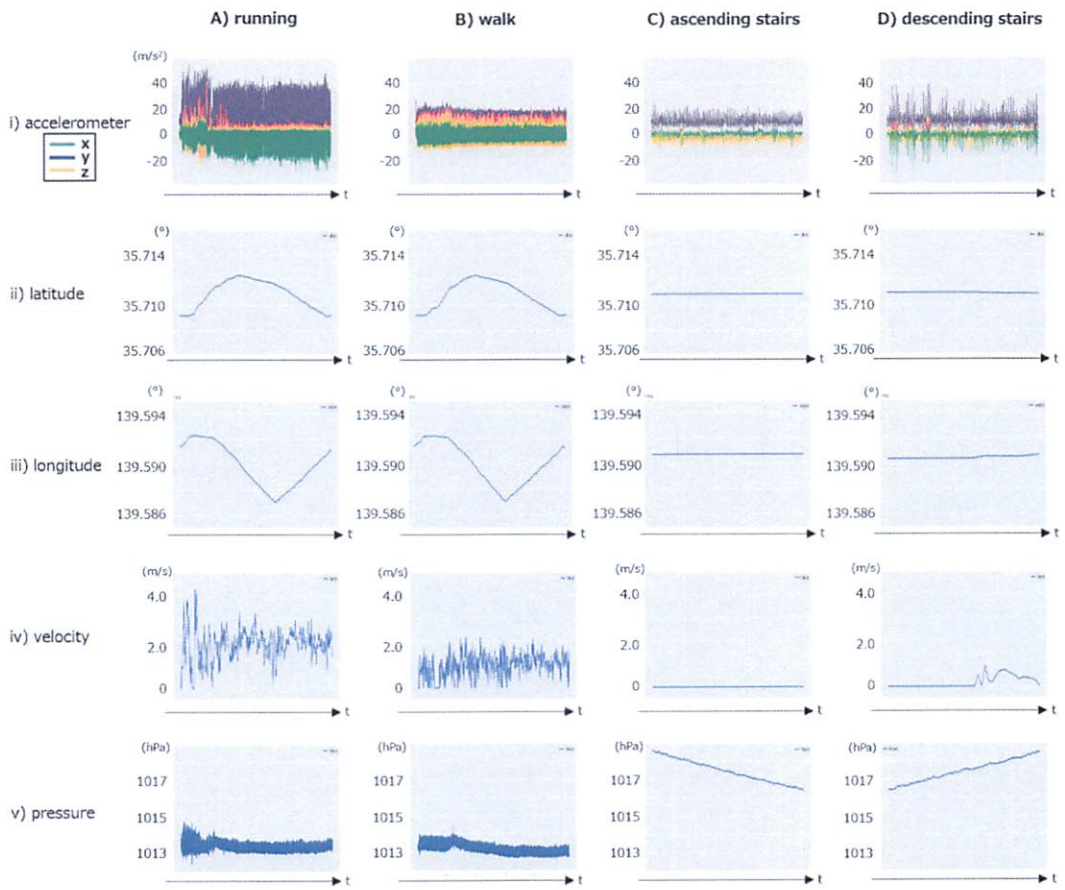

Fig 5.22. Line graph of single action.

where μ is the mean and σ^2 is the variance of the Gaussian distribution. The probability of consecutive data values missing is $\mathcal{N}(0, 200)$. Thus, the imputation target area comprises 1,694 rows in x, 1,359 rows in y, and 1,285 rows in z.

We measured the sensor data using phyphox [48], which is a smartphone application. This application acquires sensor data using functions that are built into the smartphone, such as the accelerometer, magnetometer, gyroscope, light sensor, pressure sensor, proximity sensor, microphone, and location using GPS. The accelerometer data are composed of data along the three axes x, y, and z in units of $\mathrm{m/s^2}$. The location data from the satellite are composed of the latitude, longitude, height [m], velocity [m/s], direction [°], horizontal accuracy [m], and vertical accuracy [m]. The pressure data are composed of only pressure [hPa] designed to determine the vertical

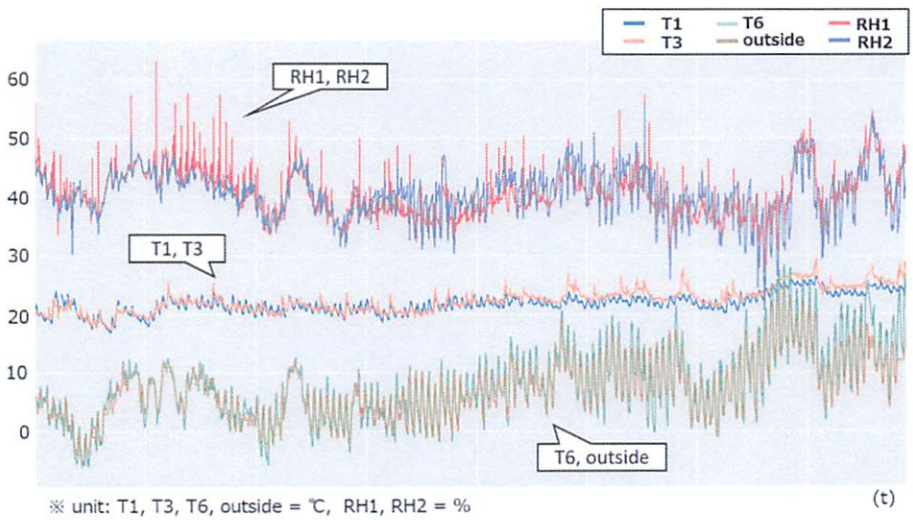

Fig 5.23. Line graph of weather data. ©2020 IEEE in literature [9]

location of the user within a building; approximately 0.1 hPa = 0.1 mbar.

Temperature and Humidity Data We used two datasets for the weather data. The first dataset contains the temperature and humidity sensor data and the other dataset contains the weather data of the local area collected by the temperature and humidity sensor.

The dataset containing temperature and humidity is composed of data collected by wireless sensors (DHT-22) installed inside or outside a home [43]. This dataset contains 29 columns, for example, measurement time, temperature, humidity, pressure, and wind speed. The temperature and humidity data were acquired by nine sensors installed on the first floor, second floor, and outside. The time span of the original dataset was 137 days (4.5 months) with 19,735 rows of data per sensor. Each sensor transmitted data approximately once every 3.3 min and the data were then aggregated from 3.3 min to 10 min. The digital DHT-22 sensors have an accuracy of $\pm$ 0.5°C for temperature measurements and $\pm$ 3% for relative humidity. Although the original data occupied 29 columns, we extracted $T1$, $T3$, $T6$, $RH1$, and $RH2$ for the experiment, where $T1$ is the temperature in the kitchen, $T3$ is the temperature in the laundry, $T6$ is the temperature outside the building, $RH1$ is the humidity in the kitchen, and $RH2$ is the humidity in the living room. These data are plotted in Fig. 5.23. In the figure, outside is the temperature from the dataset of the weather data of the local area. In this experiment, $T1$, $T6$, and $RH1$ are the inference data that is the target imputation data, and $T3$ is the training data for $T1$, *outside* is that for $T6$, and $RH2$ is that for

Table 5.16. Imputation models $\mathcal{M}$ in human activity data.

m_1	Fbprophet generated from running data
m_2	Fbprophet generated from walking data
m_3	Fbprophet generated from ascending stairs data
m_4	Fbprophet generated from descending stairs data
m_5	GRU generated from running data
m_6	GRU generated from walking data
m_7	GRU generated from ascending stairs data
m_8	GRU generated from descending stairs data
m_9	Spline interpolation

Table 5.17. Imputation models $\mathcal{M}$ in temperature and humidity data.

m_1	Fbprophet generated from sunny data
m_2	Fbprophet generated from cloudy data
m_3	Fbprophet generated from rainy data
m_4	Fbprophet generated from snowy data
m_5	GRU generated from sunny data
m_6	GRU generated from cloudy data
m_7	GRU generated from rainy data
m_8	GRU generated from snowy data
m_9	Spline interpolation

$RH1$ because they are pairs of data with similar trends. Further, we configured the missing data in the target imputation data using $\lambda = 1/500$ in Eq. (5.30) and $\mathcal{N}(0, 144)$ of Eq. (5.31). Thus, we generated the following missing data: 670 rows in $T1$, 354 rows in $T6$, and 336 rows in $RH1$ as the imputation target data.

We used the local weather data as features of APREP-S. The weather dataset comprises sensor data collected in the local area of Stambruges, 24 km from the city of Mons in Belgium [46]. This dataset includes the parameters temperature, wind speed, humidity, and pressure, measured from July 1, 2008 to June 20, 2019, recorded once every hour. We chose the data relating to temperature, weather types, and humidity. These data were recorded at hourly intervals; for example, the entry corresponding to the period 1:00:00 to 1:59:59 was used as the data point at 1:00:00 in this experiment. This dataset contains 49 weather types, which we categorized into four types: sunny, cloudy, rainy, and snowy.

Experimental Settings: Imputation Model

APREP-S includes nine imputation models, of which $m_1, m_2, m_3, m_4 \in \mathcal{M}$, were generated by Fbprophet [27], a well-known GAM method published by Facebook. Four of the remaining models, m_5, m_6, m_7, and $m_8 \in \mathcal{M}$ were generated by a gated recurrent unit

(GRU) [49]. This unit is an RNN architecture that learns to encode a variable-length sequence into a fixed-length vector representation and to decode a given fixed-length vector representation back into a variable-length sequence. Each model is generated using data from the extracted weather category. The methods used to process the human activity data are listed in Table 5.16, and those that were used to process the weather data are listed in Table 5.17. $m_9 \in \mathcal{M}$ is spline interpolation.

Feature The total number of features in this experiment was eleven. Features 1–3 were used for the initial model in the model training phase, features 4, 5, 6, and 7 were used for the human activity experiment, and features 8, 9, 10, and 11 were used to evaluate the temperature and humidity data.

1. Continuous number of rows in the imputation target area.
2. Gradient between target imputation areas,

$$(v_b - v_a)/D \tag{5.32}$$

   where v_b is before the value and v_a is after the value of the target imputation area, and D denotes the number of rows in the imputation target area.
3. Gradient trend before and after the target imputation area by comparing the before area trend with the after area trend. Input "1" if both trends are positive, input "-1" if the trends oppose each other.
4. Latitude of location data.
5. Longitude of location data.
6. Velocity [m/s] of location data.
7. Pressure [hPa] data.
8. Temperature [°C] from the weather data.
9. 48 types of weather from the weather data: e.g., *clear/sunny*=113, *cloudy*=119, *moderate rainy*=302, *moderate snowy*=332.
10. Humidity [%] from the weather data.
11. Time zone of imputation targets.: "0" if the time zone is 0:00-5:59, "1" if the time zone is 6:00-11:59, "2" if the time zone is 12:00-17:59, "3" if the time zone is 18:00-23:59.

Experimental Method

We evaluated the accuracy of APREP-S by calculating the mean square error (E) of the original data and the results of APREP-S and the existing methods. A smaller E indicates a higher accuracy.

$$E = \frac{1}{N}\sum_{d=1}^{D}(org_d - v_d)^2 \tag{5.33}$$

where D is the number of target imputation rows, org is the original value, and v_d is the value according to APREP-S or the existing methods it is being compared to. Four existing methods of imputation were used for comparison in this experiment: 1) Fbprophet as a representative GAM, 2) GRU as a representative RNN, and 3) spline interpolation as a representative of a single imputation. Fbprophet was generated from the inference data extracted from the first 100.000 rows of human activity data, whereas it was generated from all the inference data relating to temperature and humidity, for example, $T1$. GRU was generated from all inference data in both the human activity data and temperature and humidity data.

Experimental Procedure

In the model training phase, APREP-S generates imputation models from Fbprophet and GRU, and it generates an initial APREP-S model. In the temperature and humidity data, the training data were obtained by extracting the first 100,000 rows. The configuration of Fbprophet is as follows: mode is "additive," period is 1 s, and the Fourier order is 5. GRU was KerasAPI from TensorFlow. The configuration is as follows: the step size is 50 and the batch size is 100. The number of hidden units was 200, and the number of training iterations was 100 in the human activity data. In the temperature and humidity data, the configuration was as follows: the step size was 18 and the batch size was 32. The number of hidden units was 200, and the number of training iterations was 200. Spline interpolation is an interpolate library of the SciPy API. Moreover, in the training algorithm of the APREP-S model, the two parameters α and β depend on the Gaussian distribution $\mathcal{N}(0, 2)$. We used the PyMC3 library and configured NUTS as a step method with a step number of 4,000.

In the model updating phase, APREP-S provides the cluster by HMM, a GaussianHMM of the hmmlearn library. The number of clusters is eight, which is twice the number of methods, the covariance type is full, and the maximum number of iterations is 300.

In this experiment, we compared the accuracy of the imputation values when the model was updated once after the initial model generation.

Table 5.18. Result of E by mean square error (Eq. (5.33)) in human activity data.

Inference data	Training data	Imputation area	APREP-S	Fbprophet	GRU	Spline
x	each type	1,694 rows	5.51	3.00	4.43	4.44
y	each type	1,359 rows	7.33	6.07	7.90	8.61
z	each type	1,285 rows	6.74	5.70	9.00	7.95

* The unit is m/s^2
* each type of activity data means only running, walking, ascending up, and descending stairs.

Table 5.19. Result of E by mean square error (Eq. (5.33)) in temperature and humidity data.

Inference data	Training data	Imputation area	APREP-S	Fbprophet	GRU	Spline
$T1$	$T3$	670 rows	0.83	1.03	2.88	1.01
$T6$	outside	354 rows	2.02	2.36	6.46	2.46
$RH1$	$RH2$	336 rows	2.95	2.90	4.97	2.30

* The temperature unit is °C, humidity unit is %.

Experimental Result and Discussion

The results of the accuracy of the imputation values by the mean squared error are listed in Table 5.18 and Table 5.19. In the human activity data, the smallest E was obtained for Fbprophet along all axes. However, the calculation of DTW [44], which detects patterns in a data stream or time-series by measuring the distance between the data, showed that the trend among the data determined by Fbprophet is not similar to that of the original data. The results are presented in Table 5.20. These values are the distance from the original data; thus, a smaller result indicates a similar trend. The result obtained with Fbprophet is worse than that of APREP-S, except for the result for the x-axis, which is affected by the variance and standard deviation of the data. These results are provided in Table 5.22, which indicates that the value for the x-axis is the smallest.

When processing the temperature and humidity data, APREP-S yields the smallest E on the temperature data; however, the spline produces the smallest E and APREP-S the second smallest E on the humidity data. However, comparing the original data with APREP-S, Fbprophet, and spline by DTW, and the trend obtained with APREP-S is more similar to that of the data than the spline. The results of DTW are listed in Table 5.21.

We consider APREP-S as an outstanding analysis method because of its ability to accommodate data with several different periodicities. The results confirmed that APREP-S can accurately infer two types of imputation data: 1) human activity data as data with short periodicity, and 2) temperature and humidity data as data with long periodicity.

Table 5.20. Result of DTW comparing original data with human activity data.

Inference data	APREP-S	Fbprophet
x	15684.31	11438.66
y	8073.69	38936.20
z	27745.23	43225.84

Table 5.21. Result of DTW comparing original data with temperature and humidity data.

Inference data	APREP-S	Fbprophet	Spline
$RH1$	467.23	797.21	515.80

The summary of the results of this experiment is as follows.

- We verified that APREP-S can generate a suitable method using data with various periodicities by using human activity data and temperature and humidity data as examples.
- We compared APREP-S with an existing method using only one imputation method, and verified that APREP-S is more accurate by calculating the mean squared error.

5.4.5 Evaluation Result and Discussion

In the experiments, we evaluated the accuracy and the trend similarity to the original data. It can be concluded that APREP-S can infer short- and long-term periodicity data to handle outliers and missing data. To train APREP-S, we used both data with a similar trend to the target imputation data, and own-data. Moreover, in the model-update phase, APREP-S can improve inference accuracy through site-specific features, which can be input by the analyst using the PBE approach and HMM. A summary of the results is shown in Table 5.23. In Experiments 1 and 2, it was demonstrated that the ability of APREP-S to select the optimal imputation models depends on the features of the target imputation area. The size of the target imputation area was 1 or more, and the training data had a similar-trend to that of the target imputation data. In Experiment 3, we used the target imputation data and HMM to update the model. Moreover, we verified the site-specific features when the APREP-S model was updated. In Experiment 4, several imputation models were generated from an imputation method to train more suitable models using multiple types, such as periodicity or action.

Table 5.22. Variance and standard deviation of accelerometer in human activity data.

Data	Variance	Standard deviation
x	8.32	2.88
y	21.63	4.65
z	14.97	3.87

Table 5.23. Evaluation result.

Aspect	APREP-S
Target imputation data	short- and long-term periodicity
Training data	Similar-trend data and own-data
Features	Different features between model-generating site and project site
Update	HMM to support the update process
Imputation model	multiple models generated from one imputation method

5.5 Summary

We proposed a new imputation method, APREP-S. A major advantage of APREP-S is its ability to define multiple models for each specific imputation method and select the optimal model for each target imputation area. In addition, based on the PBE approach in the model-updating phase, the analyst can update the APREP-S model using site-specific features. This is effective when the model-operating site, that is, the project site, is different from the model-generating site.

The summary of this chapter are as follows:

- We proposed a new imputation method for outliers and missing data based on machine learning integrated with human knowledge using a PBE approach to reduce the analysis resources required by IT engineers.
- It was demonstrated that APREP-S can improve accuracy by selecting the optimal imputation models and by configuring the features for the analysis. Short- and long-term periodicity data were used as the target imputation data, and it was demonstrated that it is equally effective to use training data with a similar-trend to the target imputation data or own-data.
- We compared APREP-S with existing imputation methods in terms of accuracy and trend similarity to the original data.

Chapter 6

Conclusion and Future Works

6.1 Conclusion

In this study, we proposed the data mining framework APREP-DM that includes the data imputation method APREP-S. Integrating the data mining framework and data imputation method enables preprocessing in various imputation areas and model customization using human knowledge. In the pre-processing step, APREP-DM defines two substeps: 1) a common preprocessing process, and 2) an additional preprocessing process. We found that the pre-processing can operate automatically defining the business understanding beforehand, by evaluation APREP-DM using the scenario-based and qualitative evaluation. The overview of APREP-DM and APREP-S is shown in Fig. 6.1. As the data imputation method for outliers and missing data, APREP-S, can select the most optimal imputation model from multiple methods. It calculates the probability of each model in the target imputation area based on Bayesian inference. In the model-operation phase, APREP-S works automatically using the model. In the model-updating phase, APREP-S updates the model using the PBE approach with HMM. We found that APREP-S is a suitable method or pre-processing sensor data.

The conclusions of this study are as follows.

- The automated preprocessing process can be defined by clarifying the business understanding before pre-processing. APREP-DM allows the analyst to reduce the amount of analytical resources used.

- Using APREP-S for data imputation during pre-processing in APREP-DM helps analysts with few IT skills. APREP-S selects the optimal imputation models from multiple data imputation methods defined in APREP-S beforehand. In addition, APREP-S maintains the accuracy of the model, even if the model-operating site and updating site are different from the model-generating site.

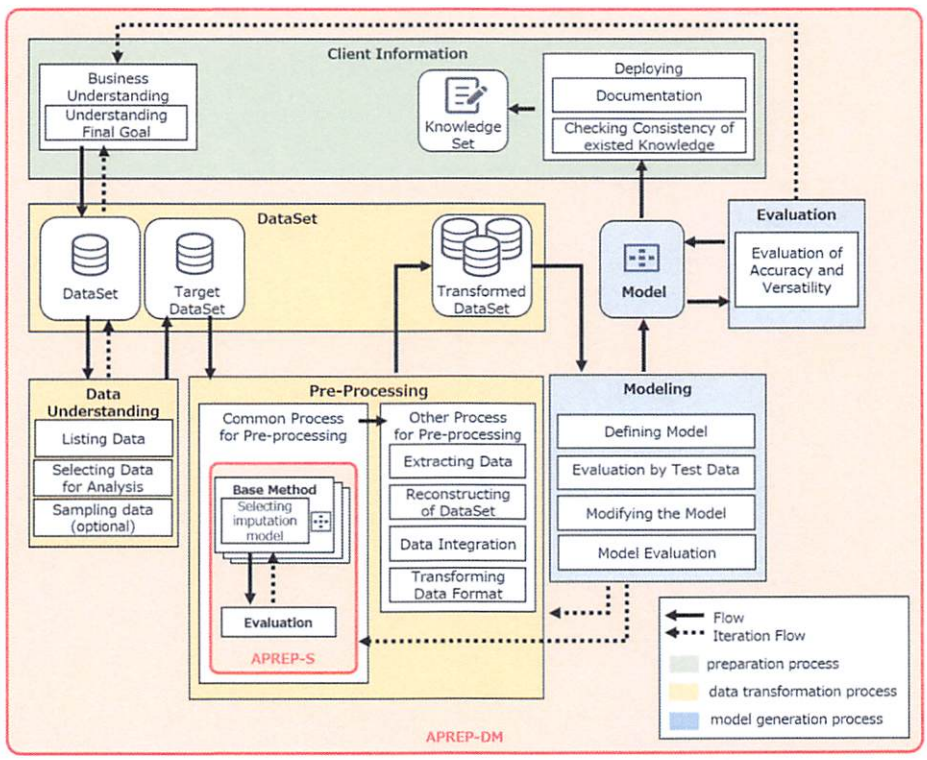

Fig 6.1. Proposed framework and imputation method of this thesis.

- The evaluation of APREP-DM and APREP-S by comparison with well-known frameworks and data imputation methods showed that APREP-DM is a well-balanced framework, and that APREP-S is an efficient imputation method for reducing and supporting data analysis compared with existing methods.

6.2 Future Works

In this thesis, we focused on time-series data, especially sensor data. We verified that APREP-DM and APREP-S are efficient pre-processing methods. However, APREP-S must be enhanced when adopting data with multiple time-series. sFor example, with the current APREP-S method, if there are multiple time-series of a person's ID data in the dataset, then that person's ID data must be extracted before it runs. Moreover, we must consider whether the range of the automatic process has room for enhancement, such as the number of clusters for HMM in the model-updating phase. The automatic process can reduce the process more than manual; however, the length of the manual

process increases as the number of clusters increase. In future work, we hope to utilize the machine learning model for clustering multiple time-series data as the target of the APREP-S imputation method.

Acknowledgements

I am deeply grateful to the people who helped and supported me during my Ph. D. studies. First, I would like to express my gratitude to my supervisor, Professor Yuka Kato at Tokyo Woman's Christian University, who has continuously supported my Ph. D. studies through valuable discussions, comments, immense knowledge of my related research, and warm encouragement. I would like express my appreciation to Professors Takeshi Ogita and Toshinori Oaku at Tokyo Woman's Christian University for providing feedback on my research, and to Professor Takako Hashimoto for his generous support and encouragement. Finally, I would like to thank the staff at Tokyo Woman's Christian University, all the group members in my corporation, and my family. Without their encouragement and persistent help, this dissertation would not have been possible.

References

[1] Fayyad, Usama and Piatetsky-Shapiro, Gregory and Smyth, Padhraic, "From Data Mining to Knowledge Discovery in Databases," *AI Magazine*, vol. 17, no. 3, pp. 37–37, 1996, https://www.aaai.org/ojs/index.php/aimagazine/article/view/1230 (refdate:2020-09-21).

[2] Hiroko Nagashima and Yuka Kato, "APREP-DM: a framework for automating the Pre-Processing of a sensor data analysis based on CRISP-DM," in *PerFoT'19 - International Workshop on Pervasive Flow of Things (PerFoT'19)*, Kyoto, Japan, 5 2019, pp. 555–560.

[3] Cross Industry Standard Process for Data Mining Consortium, "CRISP-DM by Smart Vision Europe," http://crisp-dm.eu/reference-model/, Cross Industry Standard Process for Data Mining Consortium, (refdate:2019-12-14).

[4] SAS Institute Inc., "SAS Enterprise Miner," https://web.archive.org/web/20120308165638/http://www.sas.com/offices/europe/uk/technologies/analytics/datamining/miner/semma.html/, SAS Institute Inc., (refdate:2020-06-13).

[5] IBM Corporation, "ASUM-DM Teaser," http://gforge.icesi.edu.co/ASUM-DM_External/index.htm#cognos.external.asum-DM_Teaser/deliveryprocesses/ASUM-DM_8A5C87D5.html, IBM Corporation, (refdate:2020-06-14).

[6] Hiroko Nagashima, Yuka Kato, "Method for Selecting a Data Imputation Model Based on Programming by Example for Data Analysts," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, (December, 2020 in press).

[7] D. Brscic, T. Kanda, T. Ikeda, and T. Miyashita, "Person Tracking in Large Public Spaces Using 3-D Range Sensors," *IEEE Transactions on Human-Machine Systems*, vol. 43, pp. 522–534, 2013, http://ieeexplore.ieee.org/document/6636027/ (refdate:2020-09-21).

[8] Hiroko Nagashima, Yuka Kato, "Data Imputation Method based on Programming by Example: APREP-S," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 4412–4421.

[9] Hiroko Nagashima and Yuka Kato, "Recommendation of Imputing Value for Sensor Data based on Programming by Example," *Journal of Information Processing*, vol. 28, pp. 102–111, 2020.

[10] S. Gulwani and P. Jain, "Programming by examples: PL meets ML," in *Programming Languages and Systems*, B.-Y. E. Chang, Ed. Springer International Publishing, 2017, vol. 10695, pp. 3–20. [Online]. Available: http://link.springer.com/10.1007/978-3-319-71237-6_1

[11] Qi, Zhixin and Wang, Hongzhi and Li, Jianzhong and Gao, Ho, "Impacts of dirty data: and experimental evaluation," *arXiv:1803.06071 [cs, stat]*, 3 2018.

[12] Gulwani, Sumit and Harris, William R. and Singh, Rishabh, "Spreadsheet Data Manipulation Using Examples," *Commun. ACM*, vol. 55, no. 8, pp. 97–105, 8 2012, http://doi.acm.org/10.1145/2240236.2240260 (refdate:2020-09-21).

[13] Jin, Zhongjun and Anderson, Michael R. and Cafarella, Michael and Jagadish, H. V., "Foofah: Transforming data by example," in *Proceedings of the 2017 ACM International Conference on Management of Data*, ser. SIGMOD '17. ACM, 2017, pp. 683–698. [Online]. Available: http://doi.acm.org/10.1145/3035918.3064034

[14] U. Naohiko, "DX (Digital Transformation) : 7. The Present and Future of Digital Transformation (DX)," *Information Processing Society of Japan*, vol. 61, no. 11, oct 2020, (in Japanese).

[15] Siemens, "Transforming into a Digital Company," https://new.siemens.com/jp/en/company/topic-areas/casestudy-konicaminolta.html, Siemens, (refdate:2020-06-06).

[16] M. Akihiko, "DX (Digital Transformation) :8. Digital Transformation in Airline Industry Increasing Customer and Employee Satisfaction with Digital Technology," *Information Processing Society of Japan*, vol. 61, no. 11, oct 2020, (in Japanese).

[17] Lee, Jay and Bagheri, Behrad and Kao, Hung-An, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 3, pp. 18–23, 2015, https://linkinghub.elsevier.com/retrieve/pii/S221384631400025X(refdate:2020-05-03).

[18] de Reuver, Mark and Sorensen, Carsten and Basole, Rahul C., "The Digital Platform: A Research Agenda," *Journal of Information Technology*, vol. 33, no. 2, pp. 124–135, 2018, publisher: SAGE Publications Ltd. [Online]. Available: https://doi.org/10.1057/s41265-016-0033-3

[19] Fayyad, Usama and Piatetsky-Shapiro, Gregory and Smyth, Padhraic, "The KDD Process for Extracting Useful Knowledge from Volumes of Data," *Commun. ACM*, vol. 39, no. 11, pp. 27–34, 1996, http://doi.acm.org/10.1145/240455.240464 (refdate:2020-09-21).

[20] Angee, Santiago and Lozano-Argel, Silvia I. and Montoya-Munera, Edwin N. and Ospina-Arango, Juan-David and Tabares-Betancur, Marta S., "Towards an Improved ASUM-DM Process Methodology for Cross-Disciplinary Multi-organization Big Data & Analytics Projects," in *Knowledge Management in Organizations*, ser. Communications in Computer and Information Science, Uden, Lorna and Hadzima, Branislav and Ting, I-Hsien, Ed. Springer International Publishing, 2018, pp. 613–624.

[21] Graham, John W., "Missing data analysis: Making it work in the real world," *Annual Review of Psychology*, vol. 60, no. 1, pp. 549–576, 2009, http://www.annualreviews.org/doi/10.1146/annurev.psych.58.110405.085530 (refdate:2019-03-07).

[22] Pedersen, Alma B and Mikkelsen, Ellen M and Cronin-Fenton, Deirdre and Kristensen, Nickolaj R and Pham, Tra My and Pedersen, Lars and Petersen, Irene, "Missing data and multiple imputation in clinical epidemiological research," *Clinical Epidemiology*, vol. 9, pp. 157–166, 3 2017, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5358992/ (refdate:2019-05-03).

[23] Mckinley, Sky and Levine, Megan, "Cubic spline interpolation," *Coll. Redw.*, vol. 45, 2 1999.

[24] Qi Jacky Cui, Thad Guidry, Martin Magdinier, "Openrefine," http://openrefine.org/, Supported by Google News Initiative, (refdate:2019-05-05).

[25] Trifacta, "Trifacta wrangler," https://www.trifacta.com/start-wrangling/, Trifacta, (refdate:2019-05-05).

[26] Hastie, Trevor and Tibshirani, Robert, "Generalized additive models," *Statistical Science*, vol. 1, no. 3, pp. 297–310, 1986, https://www.jstor.org/stable/2245459 (refdate:2019-07-24).

[27] Taylor, Sean J and Letham, Benjamin, "Forecasting at scale," in *PeerJ Preprints*, 2017, https://peerj.com/preprints/3190 (refdate:2019-07-15).

[28] Hochreiter, Sepp and Schmidhuber, Jurgen, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, https://doi.org/10.1162/neco.1997.9.8.1735 (refdate:2019-07-15).

[29] R.A. Fisher, Michael Marshall, "UCI Repository of Machine Learning Databases - Iris Data Set - ," https://archive.ics.uci.edu/ml/datasets/Iris, (refdate:2020-06-28).

[30] Paparrizos, John and Gravano, Luis, "k-shape: Efficient and accurate clustering of time series," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data - SIGMOD '15*. ACM Press, 2015, pp. 1855–1870. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2723372.2737793

[31] A. Cypher and D. C. Halbert, *Watch what I Do: Programming by Demonstration*. MIT Press, 1993.

[32] Gulwani, Sumit, "Automating string processing in spreadsheets using input-output examples," in *Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, ser. POPL '11. ACM, 2011, pp. 317–330. [Online]. Available: {http://doi.acm.org/10.1145/1926385.1926423}

[33] Kini, Dileep and Gulwani, Sumit, "Flashnormalize: Programming by examples for text normalization," in *IJCAI*, July 2015, pp. 776–783. [Online]. Available: https://www.microsoft.com/en-us/research/publication/flashnormalize-programming-examples-text-normalization/

[34] Menon, Aditya Krishna and , Omer Tamuz and Gulwani, Sumit and Lampson, Butler and Kalai, Adam Tauman, "A machine learning framework for programming by example," in *Proceedings of the 30th International Conference on Machine Learning (ICML), 2013*, vol. 28, no. 1, June 2013, pp. 187–195. [Online]. Available: https://www.microsoft.com/en-us/research/publication/machine-learning-framework-programming-example/

[35] Azevedo, Ana Isabel Rojao Lourenco and Santos, Manuel Filipe, "KDD, SEMMA and CRISP-DM: a parallel overview," *IADS - DM*, pp. 182–185, 2008, https://recipp.ipp.pt/handle/10400.22/136 (refdate:2020-09-21).

[36] Ministry of Land, Infrastructure, Transport and Tourism, "Japan meteorological agency," https://www.data.jma.go.jp/gmd/risk/obsdl/index.php, (refdate:2020-06-27).

[37] Tsochantaridis, Ioannis and Joachims, Thorsten and Hofmann, Thomas and Altun, Yasemin, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005, https://www.jmlr.org/papers/v6/tsochantaridis05a.html(refdate:2020-09-21).

[38] Talend, "Talend Open Studio," https://www.talend.com/, Talend, (refdate:2020-10-03).

[39] Hiroko Nagashima and Yuka Kato, "Flexible Imputation Method for Sensor Data based on Programming by Example: APREP-S," *Journal of Information Processing*, vol. 29, 2021, (February 2021 in press).

[40] Bishop, Christopher M., *Pattern recognition and machine learning*, ser. Information science and statistics. Springer, 2006.

[41] Davidson-Pilon, Cameron, *Bayesian Methods for Hackers: Probabilistic Programming and Bayesian Inference*, 1st ed. Addison-Wesley Professional, 5 2015.

[42] Cover, T. and Hart, P., "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967, http://ieeexplore.ieee.org/document/1053964/ (refdate:2019-12-15).

[43] Candanedo, Luis M. and Feldheim, Veronique and Deramaix, Dominique, "Data driven prediction models of energy use of appliances in a low-energy house," *Energy and Buildings*, vol. 140, pp. 81–97, 04 2017, https://linkinghub.elsevier.com/retrieve/pii/S0378778816308970 (refdate:2019-03-09).

[44] Berndt, Donald J. and Clifford, James, "Using dynamic time warping to find patterns in time series," in *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, ser. AAAIWS'94, vol. 03. AAAI Press, 4 1994, pp. 359–370. [Online]. Available: http://dl.acm.org/citation.cfm?id=3000850.3000887

[45] Keogh, Eamonn J. and Pazzani, Michael J., "Derivative Dynamic Time Warping," in *Proceedings of the 2001 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, 4 2001, pp. 1–11.

[46] World Weather Online, "World weather online," https://www.worldweatheronline.com/, Facebook, (refdate:2019-07-15).

[47] Facebook's Core Data Science team, "Prophet," https://facebook.github.io/prophet/, Facebook, (refdate:2019-07-15).

[48] Sebastian Staacks, "phyphox - physical phone experiments," https://phyphox.org/, 2nd Institute of Physics of the RWTH Aachen University, (refdate:2020-04-24).

[49] Cho, Kyunghyun and van Merrienboer, Bart and Gulcehre, Caglar and Bahdanau, Dzmitry and Bougares, Fethi and Schwenk, Holger and Bengio, Yoshua, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," *arXiv:1406.1078 [cs, stat]*, 2014, http://arxiv.org/abs/1406.1078 (refdate:2019-12-18).

[50] M. J. Barrenechea and T. Jenkins, *DIGITAL MANUFACTURING*, 1st ed. Open Text Corporation, 2018.

[51] 清野 武寿, 池田 義雄, "デジタルマニュファクチャリングによるモノづくり変革-Digital Manufacturing for Innovative Manufacturing Management and Engineering-," 東芝レビュー, vol. 58, no. 7, p. 6, 2003, https://www.toshiba.co.jp/tech/review/2003/07/58_07pdf/a02.pdf (refdate:2020-09-22).

[52] "NEC industrial IoT - for manufacturing in the age of IoT : NEC technical journal," https://www.nec.com/en/global/techrep/journal/g17/n01/170107.html, NEC, (refdate:2020-06-06).

Appendix A

Use Case of Digital Manufacturing

Cyber-physical systems are defined as transformative technologies for managing interconnected systems between their physical assets and computational capabilities. Digital transformation occurs on digital platforms. This digital revolution leads technology advancements in software analysis, including advancements in machine learning, quantum mechanics, robotics, Internet of Things (IoT), material science, and automated cars. Three quarters of manufacturing companies believe that digital transformation is a revolutionary opportunity [50]. In recent years, analysis connecting various data is increasing being implemented, and companies have provided solutions and products for analysis them. The concept of data connection has existed for more than 20 years. For example, Toshiba Corporation proposed increasing manufacturing speed with IT within a framework for efficient flow: the concept of digital manufacturing [51]. NEC developed NEC Industrial IoT [52] for digitizing on-site information by implementing an IoT system on the production line and standardizing the manufacturing system.

We describe one use case. Konica Minolta, Inc., has challenged the one-stop provision of IoT solutions, from consulting to operations, through visualization [15]. Its the digital manufacturing outside sales business deploys advanced image processing and digital technology and has evolved into its core businesses. This business supports a one-stop platform by visualizing, analyzing, and processing the movement of people, things, and equipment in plants of regional manufacturers. In the future, they will connect the analysis data not only at individual plants, but also at all of the connected plants.

Publication Related to this Dissertation

- Hiroko Nagashima, Yuka Kato, "APREP-DM: a framework for automating the Pre-Processing of a sensor data analysis based on CRISP-DM," IEEE Pervasive Computing and Communications 2019 Workshop (Pervasive Flow of Things 2019), pp. 555–560, 2019 [Chapter4]
- Hiroko Nagashima, Yuka Kato, "Recommendation of Imputing Value for Sensor Data based on Programming by Example (February, 2020)," Journal of Information Processing vol. 28, pp. 102–111, 2020 [Chapter5]
- Hiroko Nagashima, Yuka Kato, "Flexible Imputation Method for Sensor Data based on Programming by Example: APREP-S," Journal of Information Processing vol. 29, February 2021 in press. [Chapter5]