RTLS-enabled clinical workflow predictive analysis

Zhang, M.

Award date:
2017

Technische Universiteit Eindhoven University of Technology
Department of Mathematics and Computer Science

RTLS-Enabled Clinical Workflow Predictive Analysis

Ming Zhang

in partial fulfillment of the requirements for the degree of
Master of Science in **Business Information Systems**

Supervisors:
Prof. Dr. Mykola Pechenizkiy
Dr. Supriyo Chatterjea

Assessment committee members:
Prof. Dr. Mykola Pechenizkiy
Dr. Supriyo Chatterjea
Dr. Dirk Fahland
Dr. Ir. Irene Vanderfeesten

Eindhoven, July 21, 2017
Published on July 21, 2018

Abstract

Workflow management and analysis in clinical settings have been studied extensively. However, due to the limitations of cost and data sources, traditional methods of recording and analyzing workflow events, especially for a long period of time in the hospitals in large scales are not feasible. In addition, currently it is not possible to track workflow events automatically and continuously using existing hardware and software technologies.

In recent years, Real-time Location Systems (RTLS) has became more and more popular among hospitals. Current application scope of RTLS is still narrow. However, RTLS can potentially be a great tool to breakthrough the current limitations in workflow analysis as it can provide accurate indoor location tracking for different types of entities and produce streaming location data in real-time. This streaming data has great potential in many applications of workflow analysis and prediction.

In this thesis, a set of novel approaches for analyzing, identifying and predicting clinical workflow events using the combination of Machine Learning and Real-time Location Systems (RTLS) is introduced. Using this combination, we tackle four clinical use cases: (1) Create a dynamic visualization to playback historical workflows by displaying animations of tag movements; (2) Analyze the location sequence of staff tags to find patterns in particular workflow events; (3) Automatically identify workflow events using RTLS data; (4) Predict the remaining time for ongoing workflow events.

The four use cases are converted into three data analytics tasks, including map and parallel coordinates visualization, machine learning based classification and regression.

The thesis is focused on how to properly train machine learning, especially deep learning models using historical RTLS, machine log and Case Report Form (CRF) data to give classifications and predictions purely based on future RTLS data. The models will be compatible to make online predictions in real clinical settings where we do not have other data sources.

The evaluation result collected from a series of experiments shows that the proposed approach of automatically identifying workflow events is feasible. The deep learning model: LSTM based on sequential RTLS data yields 0.87 in AUC, 0.63 in Kappa and 0.84 in accuracy on test data. The approach of automatically predicting the remaining time of ongoing event yields 14 minutes Mean Absolute Error (MAE) in predicting the remaining time of exam events which in average last about 3 hours. We conclude that our models are applicable in automatically detecting workflow events and predicting event remaining time in real-time in clinical environments where staffs are tracked by RTLS.

Acknowledgment

I would like to express my profound gratitude to my company supervisor Dr. Supriyo Chatterjea. He has provided me this valuable opportunity of solving real-world business puzzles with real data in a company setting. During the project, he has given me firm support in providing data and chances to talk with domain experts. He has provided valuable suggestions and encouragements. His guidance guaranteed the project to be always on the right track.

I would like to sincerely thank my university supervisor Prof. Dr. Mykola Pechenizkiy for his full support in my two years of master's degree study and in this graduation project. Prof. Pechenizkiy has introduced me the opportunity to do an internship and graduation project in Philips Research. He always gives me the room to improve my professional skills in my own way while providing critical suggestions and continuous encouragement.

Nevertheless, I would like to thank Dr. Dirk Fahland and Dr. Ir. Irene Vanderfeesten for joining the assessment committee. Dr. Fahland is my first year mentor, he has given me enormous help in starting my journey in BIS and in choosing the stream of my study. Dr. Ir. Vanderfeesten has given me impressive lectures in process quality assessment and redesign in my first year of study.

I would like to acknowledge Prof. Dr. Ir. Evert van Loenen for his intensive support in deploying and configuring the RTLS environment. He has provided valuable suggestions for improving the quality and reliability of the RTLS data.

I would also like to offer my heartfelt gratitude to my girlfriend Karolina Abratańska and my family for always standing behind me and giving me unfailing spiritual support throughout my study.

Finally, I would like to thank my colleagues: Corné, Yan, Timen, Sakina, Aaqib, Alessio, Nathan and Ilyas for spending the unforgettable time with me and giving me precious help during my graduation project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This master thesis project is carried out within Data Science group of Philips Research Eindhoven and Data Mining group of the Mathematics and Computer Science department of Technical University of Eindhoven (TU/e). The thesis is for the Business Information Systems master at TU/e.

Firstly in this chapter, the business context is discussed to motivate our research. Secondly, the identified business use cases and related data analytics tasks are described in the research objectives section. Thirdly, the main contributions of this thesis is summarized. Lastly, the outline of this thesis is introduced.

1.1 Business Context

Workflow can be defined as the flow of work through space and time [3]. In the clinical context, workflow involves the flow of the clinical staff, patients and the information. Clinical workflow can be formed voluntarily by the clinical staff, for example physicians and nurses or can be regulated by the hospitals.

Measuring and understanding existing workflows give the management insights about the efficiency and effectiveness of the entire hospital, or of a particular department and allow them to find potential spaces to improve the workflows. Nowadays, clinical resources, including staffs, rooms and equipment are commonly limited. Effectively planning of the resources are crucial for improving hospital productivity and reducing costs.

Identifying and measuring clinical workflows can be challenging. Firstly, workflows are not always obvious and consistent. Secondly, it is usually expensive to label the workflow events. There are several commonly used conventional approaches [4] for measuring clinical staff works. In these approaches, there is usually a trade-off between cost and precision. The commonly accepted "gold standard" in labeling clinical workflow events: the direct observation method suffers from high cost and is not feasible to be applied for an indefinite period. Some cheaper methods, for instance provider interview and self-administered report result in high variation [5] and bias [4] [6] [7].

In recent years, more and more hospitals have incorporated RTLS (Real-time location systems) to monitor the locations of personnel and assets. Different from GPS (global positioning system)-like systems which are successful in outdoor locating, RTLS is capable of locating entit-

ies in complex indoor environments in real-time. Hospitals benefit from this technology in many applications [8], for instance: fast locating of medical equipment, fast locating of available clinical staff in case of medical emergencies and monitoring patient location to ensure patient safety.

However, the potential of RTLS is yet to be discovered. As a real-time and reliable source of locations of all tracked clinical resources including clinical staffs, equipment and patients, RTLS data can possibly provide insights about the real-time operational status of the hospital. In this study, we assume that the location of the clinical resources is influenced by the ongoing workflow events. We will try to exploit the RTLS data to explore whether RTLS data can enable us to visualize, identify and predict clinical workflow events.

1.2 Research Objective

In this study, we aim to combine the power of RTLS with data mining and machine learning technologies. We present novel approaches of doing predictive workflow analysis with the support of RTLS using various visualization, machine learning and deep learning algorithms to achieve intuitive visualization, workflow event identification and prediction with high precision in low cost.

We have collected data from the radiology department of a hospital in The Netherlands from different sources, including RTLS, interventional X-Ray (iXR) machine log and Case Report Form (CRF). We firstly clean and combine the data from these three sources. Secondly, we build visualization tools to visualize the relation between workflow and location in an intuitive and interactive way. Lastly, we build machine learning models to identify workflow events and predict the remaining time of ongoing workflow events based on historical and streaming RTLS data.

We assume that the existing workflow is already defined by the hospital. Thus, constructing existing workflows from the logged clinical activities or events is not one of our research objectives. All the analysis and prediction tasks will be based on the predefined workflow and events from the workflow.

1.2.1 Business Use Cases

There are two main goals of this study in the business perspective. The first goal is to validate whether RTLS can enable hospitals to better observe, analyze and optimize existing workflows. The second goal is to validate the capability of doing automated workflow event prediction by combining RTLS and machine learning. Based on the two goals, we have identified four use cases. They are listed below:

1. Workflow playback. Generating rich and interactive visualization of the workflow to intuitively show the correlation between staffs' movements and workflow events.
2. Transition sequence analysis. Generate visualization to show the transition pattern between different types of zones during a workflow event.
3. Workflow event identification. Build predictive model to automatically identify workflow events using RTLS data.
4. Event remaining time prediction. Automatically predict how long time is remaining for the ongoing workflow event, based on real-time streaming RTLS data.

The four business use cases can be categorized into two main types: offline batch analysis based on historical data and online prediction based on real-time streaming data.

1.2.2 Data Analytics Tasks

To address these four business use cases, we have identified the matching data analytics tasks. The two business use cases related to offline batch analysis can be converted into two interactive visualization tasks to help the analysts to get an intuitive idea of the tag transition patterns and correlation between workflows.

For the two business use cases related to online prediction, we propose a methodology to generate features based on RTLS data and labels based on the machine log and the CRF data, to build classification and regression models based on the generated data set to capture the correlation between staff locations and workflow events. These models are later applied on the test RTLS data to evaluate the feasibility and predictive performance. The detailed definition of the converted data analytics tasks are listed below:

1. Visualization. Visualizing dynamic indoor location and workflow data using map animations and parallel coordinates.
2. Machine learning binary classification. Using classification models to classify RTLS data to give labels on whether the RTLS information indicates an ongoing workflow event.
3. Machine learning prediction. Using prediction / regression models to give prediction on the remaining time of the ongoing workflow event based on current RTLS data.

Figure 1.1 is made to better illustrate the matching between the business use cases and the data analytics tasks. From the figure, the two business use cases related to offline analysis are combined to map and parralel coordinates visualization task. The use cases of automated workflow event identification and event remaining time prediction have their own corresponding data analytics tasks.

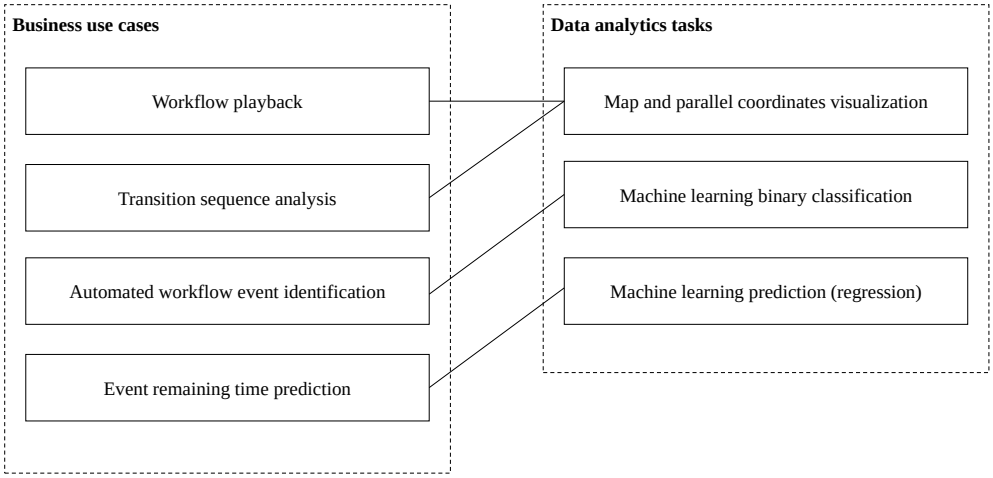


Figure 1.1: Correspondence between business cases and data analytics tasks

1.3 Main Contributions

There are four main contributions in this study: (1) I have built an interactive visualization based on D^3 which intuitively shows the relation between workflow progress and tag locations by animations; (2) I have built an interactive visualization tool based on D^3's Parallel Coordinates package to visualize the general patterns of tag transitions in different workflow events; (3) I proposed a methodology to incorporate RTLS and machine learning to automatically identify workflow events in real-time and carried out experiments to validate the methodology; (4) I proposed a methodology to predict the remaining time of ongoing workflow events automatically using RTLS and machine learning. Experiments are done to verify whether this methodology is feasible.

1.4 Thesis Outline

The context, research objective, main contributions and thesis outline are introduced in this chapter. Chapter 2 introduces the preliminary studies prior to this thesis. Chapter 3 describes the pipeline of implementing the data analytics tasks in this study. In Chapter 4, the methodologies used in cleaning the RTLS and CRF data are described. Chapter 5 introduces the framework and the implementation of visual analysis tasks including workflow playback and transitioin sequence diagram. In Chapter 6, framework, strategy and experiments of identifying workflow events using machine learning models based on RTLS are introduced. Chapter 7 describes the methodology of predicting event remaining time using machine learning models based on RTLS data and the corresponding experimental results. Lastly, Chapter 8 concludes the study outcome, describes the limitations of the study and the future works that can be done to improve this study.

Chapter 2

Background and Related Work

In this chapter, existing approaches for clinical workflow analysis related to our study are introduced. Including workflow event labeling, studies on clinical workflow analysis using data mining and process mining approaches, RTLS and its application and existing machine learning models.

2.1 Conventional Workflow Event Labeling Approaches

In this section, three conventional approaches for recording workflow events are introduced.

2.1.1 Direct Observation

In this approach, trained observers are asked to follow clinical staffs for their entire shifts, recording the behaviors of the staffs at a certain minute interval. This approach yields a high accuracy, however, the drawback is obvious: it has significant high costs, including human resource costs for observers, intrusion of clinical staffs' normal workflows [4]. It is often considered unfeasible for large scale projects [6].

In addition, a study by Kazdin [9] shows that direct observation can lead to the observer effect. An observer effect is a type of reactivity in which individuals modify an aspect of their behavior in response to their awareness of being observed [10]. This can affect the precision of direct observation.

2.1.2 Provider Interviews

In this approach, clinical staffs are given interviews at the end of their shifts. The interview includes questions regarding to their work: what types of services have they provided during the shift. For each type of the services, average duration of patient contact are estimated by the clinical staff. In addition, questions are also asked about the time the clinical staff spent on activities that are not related to direct patient care.

Although the provider interviews approach has a lower cost compared with direct observation, the result comes out of it is particularly weak as because it substantially overestimated contact time while underestimating non-productive time [4] [6].

2.1.3 Self-administered Report

Log books are distributed to clinical staffs. During the shift, a clinical staff needs to note down all the activities he or she has done in a form, as well as some descriptions about each activity. For instance activity start time, end time and a short description.

Assessments have been carried out by Hunting et al. [5] about the validity of self-administered reports. The result shows that the utilization estimated from self-administered reports is highly variable. The estimation result from self-administered reports and direct observation are compared by Burke et al. [11] . A study of Donaldson & Grant-Vallone [7] concludes that potential response bias exists in this approach, the subjects in the experiments tend to record more utilization on socially desirable activities.

For the conventional approaches introduced above, in addition to the drawbacks described, the provider interviews and self-administered report methods do not apply on estimating the utilization of clinical equipment. Also, for provider interviews, only the duration of each type of activities is recorded, we cannot know at which exact time did each activity happen and end.

2.2 Workflow Mining from Clinical Activities

One challenge for workflow analysis in hospital is to construct the workflow from recorded clinical activity logs. This is a typical process mining task. Langab [2] et al. have done a survey on varies of process mining approaches to construct clinical workflow using log data from Radiology Information System (RIS), Hospital Information System (HIS) and logs from Computer Tomography (CT), Magnetic Resonance Imaging (MR), ultrasound (US), and X-ray (XR) machines.

Various algorithms are surveyed in this study, including α algorithm introduced by van der Aalst et al. [12], the α++ algorithm by to Wen et al. [13], the heuristic mining algorithm by Weijters et al. [14], the DWS algorithm by Greco et al. [15][16], the multiphase algorithm by van Dongen and van der Aalst et al. [17], the genetic-mining algorithm by de Medeiros et al. [18], and the theory-of-regions-based algorithm by van Dongen and Busi et al. [19]

Table 2.1 shows the summary of evaluation result for the process mining algorithms listed int the previous paragraph. In the table, “+” stands for met, “-”stands for unmet and “+/-” stands for partly met.

Table 2.1: Evaluation of various process mining algorithms by Longab [2]

Assessment criteria	van der Aalst et al. (α algorithm)	Wen et al. (α++ algorithm)	Weijters et al. (heuristic-miner)	Greco et al. (DWS-algorithm)	de Medeiros et al. (genetic algorithm)	van Dongen et al (Multiphase miner)	Aalst and Rubin et al. (Region miner)
Truth to reality in contents	-	+	+	+	+/-	-	-
Noise and incompleteness	-	+/-	+	+	+	-	-
Sequences, forks, and concurrency	-	+	+	+	+	+	-
Loops	-	+	+	+	+	-	-
Repetitive activities	-	-	-	-	+/-	-	-
Fuzzy entry and end points	-	+/-	+	+	+	-	-
Process types and variants	-	-	-	+	-	-	-

2.3 Real-time Location Systems

Real-time location systems provide continuous and immediate indoor location tracking for entities, such as staff, patient and assets.

2.3.1 Hardware Infrastructure

There are two main categories of technologies to implement RTLS: Infrared (IR) based and active Radio Frequency Identification (RFID) based.

IR based localization is achieved by 3 main hardware components: monitors, tags and stars. An example of the components is shown in Figure 2.1. Monitors are placed in interesting locations, typically on the ceiling. The monitors emit IR signals to the tags, telling the tag which particular monitor it is under. Tags are equipped with IR receivers which pick up the IR signal from the monitor and convert the signal to the monitor's ID. A tag can report its status as well as the ID of the monitor to the star through radio. A tag may have different appearances and can be worn by a person or attached on equipment. For instance a badge clipped on a physician's uniform, a wristband worn on the wrist of a patient. The star can receive tags' location reports and forward the reports to the RTLS servers for streaming and storage.

(a) Centrak monitors and tags (b) Centrak Star

Figure 2.1: Example of hardware components in IR based approach

In contrast, RFID based locating is achieved by 2 components: receivers and tags. Similar to

the monitors in IR based approach, receivers are placed in interesting locations. Tags are equipped with microchips and each tag can transmit a unique signal which can be received and identified by the receivers. This is usually done with RFID (radio-frequency identification) technology. By reading the signal transmitted by the tags, a receiver can identify and report when a tag is in the location (e.g. a room) where the receiver is installed.

Each technology has its own advantages and shortcomings. In this study, we use the IR based technology mainly because it can give up room-level precision. Active-RFID based technologies, on the other hand, only give us a proximate location because the RF signal can travel through walls which can be problematic because it will sometimes be hard to decide which side of the wall is a tag actually located at.

2.3.2 Precision of RTLS

The IR based RTLS system used in this study achieves room/sub-room level precision depending on the installation of the monitors. This means we will only know in which zone is a tag located at. Within the same zone, which can be a room or a part of the room depending on the installation of monitors, it is impossible to tell where exactly a tag is located at. For instance, we can get location report Tag T_A is in room Z_A, however it is impossible to know whether T_A is standing next to the patient bed in the room or sitting in the chair next to the desk.

2.4 Machine Learning Classification and Prediction

Machine learning can be categorized into supervised learning and unsupervised learning. This thesis will involve supervised learning algorithms as we define targets for the learning tasks. We will face two supervised learning problems: classification and prediction. Classification problem is a problem where we identify which category a new data point belongs to given the machine learning model trained by training data. The prediction problem, also known as regression problem, is a type of problem where we predict a continuous value based on the features of a new data point using the machine learning model trained by training data.

2.4.1 Conventional Machine Learning Techniques

A set of conventional machine learning models is used in this study. In this section, we give a brief introduction of each conventional machine learning technique involved.

Logistic regression [20] is a special case of a regression model which gives a categorical prediction, usually a binary prediction. In this study, the prediction is a binary value, i.e. we model the conditional probability $Pr(Y = 1|X = x)$ as a function of x.

Decision tree is a tree-like structure that describes the strategy of generating a conclusion based on the data features. There are several decision tree generation algorithms. In this study, Classification And Regression Tree (CART) [21] is used. This algorithm supports both classification and regression tasks, which match the needs of this study for event identification and remaining time prediction.

Random Forest [22] is an ensemble learning algorithm for both classification and regression. It constructs multiple sub decision trees and makes predictions based on the ensemble of the sub trees. It is believed to be able to correct the over fitting problem [23].

Naive Bayes [24] is a type of simple probabilistic classifier based on Bayes' theorem with independence assumptions between the features. In this study, Gaussian Naive Bayes model is adopted.

Multilayer Perceptron is a class of feed forward artificial neural network (ANN). An multilayer perceptron consists of three or more layers of nodes. The nodes, except for the input layer, use nonlinear activation functions. Backpropagation is used for weight training [25].

Support Vector Machine [26] is a discriminative classifier which separates data according to a trained hyperplane. SVMs are especially good at non-linear classifications as it uses kernel trick which maps the inputs into high-dimensional feature spaces.

2.4.2 Deep Learning Models

In this section, two deep learning algorithms are introduced.

Recurrent Neural Networks

Recurrent neural network (RNN) is a type of artificial neural network in which the perceptrons have cyclical connections. Unlike traditional neural networks which consider the input data to be independent of each other, RNN structure enables the network to have a memory of the history, which is very useful in context-sensitive prediction tasks such as natural language processing (NLP), speech recognition and handwriting recognition. RNN network has several architectures. For instance Elman network [27] and Jordan network [28].

The drawback of RNN is that it suffers from the problem of vanishing gradient. In theory RNN can learn the dependencies between data in any intervals, but practically, it is not feasible to train a traditional RNN network to learn the long term dependencies. So usually it is only feasible able to look back in history for a few steps.

Long Short-term Memory

Long Short-term Memory (LSTM) was introduced by Hochreiter et al. in 1997 [29]. Unlike the traditional RNN architectures, LSTM does not suffer from the vanishing gradient problem introduced in the previous section. The LSTM structure is very similar to RNN, the difference is that the nonlinear units are replaced by memory blocks with recurrent gates.

Figure 2.2 depicted the structure of LSTM memory block in a cell. Different from RNN, the LSTM cell contains an input gate, an output gate and a forget gate. The input gate will decide whether to let a new input in. The output gate decides whether let the state impact the output and the forget gate decides whether to erase the cell's present state.

The advantage of this structure is that the added gates allow the network to control the status of the cells. Thus it can decide which information to discard or "forget". This makes it possible to find long term dependencies without suffering from the exploding of backpropagated errors.

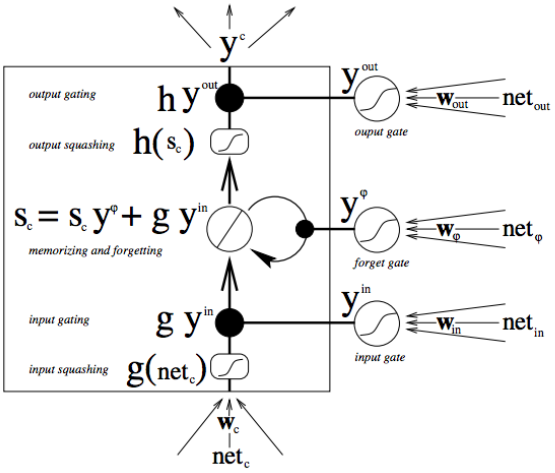


Figure 2.2: LSTM memory block structure

In our application of identifying workflow events and predicting event remaining time using RTLS data, this quality of LSTM can be of great help because it is able to look back a long period of history to find sequential patterns.

LSTM is suitable for both classification and regression tasks.

Chapter 3

Research Framework

In this chapter, the framework of implementing the data analytics tasks in this study and the steps in the framework are introduced. The framework is designed to carry out the data analytics tasks defined in Section 1.2.2 based on the business use cases introduced in Section 1.2.1.

3.1 Framework Overview

Figure 3.1 depicts the overall framework of this research study.

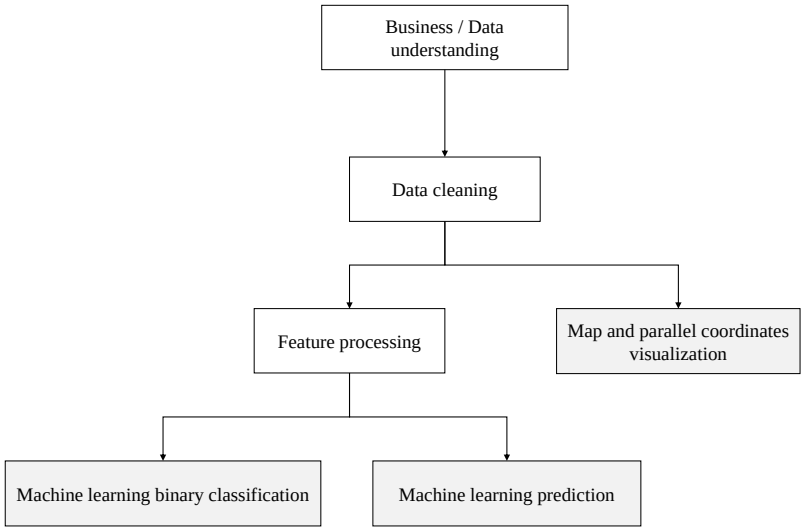


Figure 3.1: The overall framework of the implementation

The framework consists of four levels from the top to the bottom. The basic level is the understanding of the current business logic and how to relate the collected data to the business logic. The second level is to improve the data quality and convert the data in a unified format. The third level consists of three parts of work based on the cleaned data. The last level of work will be carried out based on the extracted features from the third level.

3.2 Business and Data Understanding

Understanding the business logic and data is the fundamental work in our study. This study's research object is the workflow in the interventional X-ray (IXR) section of the radiology department of a hospital. In the section, several X-ray machines are installed in separate exam rooms. We have chosen a particular exam room as the subject of this study. Thus, we study all the workflows related to the procedures that are performed in this room.

Different procedures are performed in the X-ray exam room, however, they follow the same general workflow. The workflow is predefined by the hospital.

3.2.1 Scope of the Data Set

We have collected data from 3 different sources: the RTLS data, the machine logs and the CRF data.

RTLS Data

We have collected RTLS data for 12 months. In this study, we only tagged the clinical staffs involved in the workflows. Staffs are mainly divided into 2 roles: physician and nurse. Although there is a more detailed role classification for the nurses, for instance, there can be sterile nurse and normal nurse in an exam, they will have the same type of tag so we do not know whether a tag is used by a sterile nurse or a normal nurse.

Meanwhile, we do not track individual staff to lessen privacy concerns. A staff will pick up a tag which corresponds to his/her role when his/her shift start, thus a same tag can be used by different staffs in different shifts.

Patients and assets are not tracked in this study.

iXR Machine Log

We have access to the machine log of the interventional X-Ray machine which is installed in the target exam room. The raw machine log has the following information including but not limited to:

1. UI interactions
2. Table positions and adjustments
3. Acquisitions
4. Software logs

The machine log is semi-structured. It follows certain data structure but the description of an event can be in free text. The Philips iXR team has developed a machine log cleaner to extract information from the raw machine log and generate features in different aspects into different tables. In this study, we use the exam acquisitions table which is generated by this cleaner. This table contains information regarding:

1. Exam properties

   (a) System information
   (b) Exam ID
   (c) Exam time period

2. Acquisition properties
   (a) Acquisition time period
   (b) Acquisition application type
   (c) Acquisition parameters

There can be multiple acquisitions during one exam.

Case Report Form

During the same period when the RTLS data was collected, we also collected the CRF data of the workflow procedures. The date and time of 15 types workflow events are collected for each procedure.

The 15 workflow events are listed below:

1. Patient is registered
2. Patient arrived in waiting room
3. Patient arrived in exam rom
4. Patient laid on table
5. Physician is called
6. Physician arrives in the exam room
7. Patient is clear for procedure
8. Start incision
9. Close incision point
10. Called to retrieve the patient
11. Physician leaves the exam room
12. Physician leaves the control room
13. Patient retrieved from table to bed
14. Patient dismissed
15. End of research

The workflow is patient oriented. In a procedure, only one patient is treated. Thus a procedure matches a patient visit. In other words, if a patient visits again, he will have a new procedure for this new visit.

For each procedure, one workflow event can appear at most once. Some events can be missing in a procedure. Workflow events do not necessarily follow a particular order. For example, physicians can be called before the patient is laid on the table.

In addition, several procedures may take place at the same time. A procedure does not only cover the duration of the actual exam in the exam room, but also cover the time when a patient is admitted, waiting in the waiting room etc., thus it can be the case that while one patient is waiting, the second patient is being treated in the exam and the third patient is being dismissed, i.e. there are three procedures ongoing in the same time.

3.3 Data Cleaning

Due to technical limitations of the RTLS system used, the RTLS data collected can contain missing and erroneous records. To improve the data quality and avoid making visualization and predictions based on missing and erroneous data, several data cleaning approaches are adopted to improve the quality of the RTLS data. These approaches will be introduced in Chapter 4.

The CRF data also has data quality problems. One main reason is that the input of the data is done by the nurses for the first months. This is an additional work for them while they already need to focus on the exams. Thus it can be the case that they missed to input several events, or input error events date time by mistake. Strategies are to clean the CRF data are described in Chapter 4.

The machine log data, on the other hand, does not need cleaning because cleaning is already done when we retrieve the data. Thus the cleaning for machine log data is not discussed in this thesis.

3.4 Analysis Based on Clean Data

The cleaned data is considered to be accurate and reliable. We firstly carry out visual analysis based on this data. Feature processing is also done based on the cleaned data. It will then be used in the two machine learning predictive tasks in the fourth level.

3.4.1 Visualization

The main purpose of visualization is to explore the correlation between different data sources. We develop a visualization platform to provide interactive and intuitive visualization of the historical location and workflow aligned according to the timestamp.

The scope of the visualization is only with the RTLS data and the CRF data. We did not take the machine log data into account because there is an ongoing study on visualizing the machine log data. Also, the machine log data is much more complex and detailed.

3.4.2 Feature Processing

The objective of feature processing is to prepare structured data as features and labels for the machine learning predictive tasks in the fourth level.

The reason why we cannot use the cleaned data directly as features is that the cleaned data is still in the log format. In the cleaned data, one data point contains only information about where is a tag located in a given timestamp as well as the tag and the zone properties. This data just gives us a micro point of view and this provides little value for the prediction.

Thus, in other words, the goal of feature processing is to build this data which contains the overall status of the system using the micro information from the RTLS location reports. This work is introduced in Section 6.2.

3.5 Predictive Analysis based on Machine Learning

The two tasks in the fourth level are about machine learning based predictive analysis. The purpose of this part is to address the two business use cases mentioned in Section 1.2: workflow event identification and event remaining time prediction.

3.5.1 Classification vs. Prediction

Machine learning tasks can be classified in many categories. The most popular classification is supervised learning vs. unsupervised learning [30]. Supervised learning tasks can be classified as classification or prediction tasks [30]. In classification, we use our model to predict a discrete label while in prediction we predict a continuous value. In our study, it is obvious that the event identification task is a classification task and the remaining time prediction task is a prediction task as we want to estimate a continuous remaining time.

3.5.2 Sequence learning

Most machine learning prediction algorithms, for instance logistic regression, decision tree etc. are designed for independent data. They do not consider the correlation between data points. Sequence labeling a is a type of sequence learning task. It tries to predict labels using sequences of data. It exploits the correlation between data points in that sequence to improve the classification performance [31]. Sequence learning can also be used in value prediction.

In this study, besides the classical machine learning algorithms, a deep learning based sequence learning algorithm Long Short Term Memory (LSTM) is used to mine the location sequences to achieve workflow event identification and remaining time prediction. This approach will be compared with the classical machine learning algorithms.

Chapter 4

RTLS and CRF Data Cleaning

In this chapter, we introduce the RTLS and Case Report Form (CRF) data, their respective data quality problems and our approach of cleaning RTLS and CRF data to improve the data quality.

4.1 RTLS Data Introduction

Data structure used in this study is introduced in this section. RTLS data is in the form of streaming data, which contains information of the location reports.

4.1.1 Properties of RTLS Data

RTLS data is structured data. In addition, RTLS data is time based and is sequential, the rows of data are correlated according to the time.

The time interval of RTLS data is not fixed, i.e. tags do not report their time in fixed intervals. This can be caused by:

- Tag fail to report their location to the star due to weak radio signal. This can be caused by a weak battery, block of radio signal and excessive distance between the tag and the star
- Tag has "fallen asleep". In some RTLS solutions, for instance Centrak [32], the tag will report to the star less frequently when a tag is not in motion for a certain period of time to save battery

4.1.2 Raw RTLS Data Structure

Raw RTLS data comes directly from the RTLS server. When the RTLS server receives a tag location report, a row of data is generated.

The data structure of raw location report is described in Table 4.1.

Table 4.1: Properties of raw RTLS location report

Property	Comment
Epoch time	The epoch time indicates when was this tag report generated. The time comes from the star. The Epoch time is in UNIX timestamp format, i.e. the number of seconds since 1970-01-01
Tag ID	The ID of the RTLS tag, which is an integer value
Monitor ID	The ID of the monitor this RTLS tag detected. It is also an integer
Motion status	This is a boolean field. There are motion sensors equipped in Centrak RTLS tags. When the sensor detects a movement, this field will be set as *true* in the next location report.
Button status	including boolean values for each button. If a button is pressed, then the corresponding boolean value will be *true*
Battery level	which is a boolean value. This field will be *true* if the current tag has low battery

Table 4.2: Properties of RTLS monitor metadata

Property	Comment
Monitor ID	The ID of the monitor in integer
Zone ID	The ID of the zone where the monitor is installed
Name	The name of the monitor

4.1.3 RTLS Metadata Structure

RTLS metadata can be considered as a dictionary to understand the meanings of the ID values in raw RTLS data. It also contains the configuration information of the system. It consists of the following parts:

Monitor Information

Monitor information includes monitors' hardware IDs and their configuration. The important fields are listed in Table 4.2.

The zone ID is important because it indicates us in which zone or room is a monitor installed. The monitor ID to zone ID map will be used to lookup the zone ID from the raw location report.

Zone Information

Zones are the units to measure tag locations. A zone can be a room or a section of a room when a room is divided into multiple zones. Room division is done mostly for long corridors which connect multiple rooms. It can also be done for big wards for different beds, so that the system

Table 4.3: Basic properties of a RTLS zone

Property	Comment
Zone ID	The ID of the zone in integer
Type	The type of the zone, for instance: "ward", "exam room" and "corridor"
Name	The name of the zone
Coordinate	The (x, y) coordinate of the zone. This coordinate can be the in meters or feet, or just be the pixel coordinate of the center point of the zone in the floor plan

can provide bed level accuracy.

Zone information consists of two parts:

- Basic Information: The basic information includes the properties of the zones themselves, some important properties are listed in Table 4.3.
- Zone Connectivity: From the basic zone information, we know the coordinates of the zones. However, we cannot decide the walking distance between zones based on only coordinates as there are physical walls separating rooms. Thus we need information on whether/how are different zones connected.

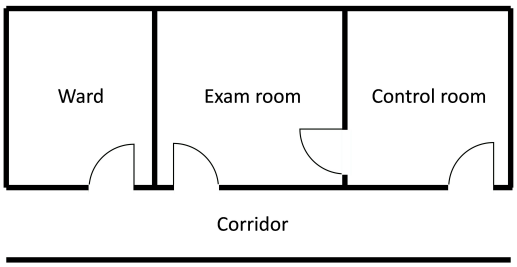


Figure 4.1: An example floor plan consists of 4 zones

Figure 4.1 shows an example of a floor plan of a part of a hospital. Suppose we have 4 zones monitored in this case: the ward, the exam room, the control room and the corridor. It is obvious that a tag cannot travel directly from the ward to the exam room as there does not exist a door which connects the two zones. To travel from the ward to the exam room, a tag needs to travel to the corridor first before it can enter the exam room. On the contrary, a tag can travel directly from the exam room to the control room, as there exists a door connecting these two rooms.

This information is important for the lateral workflow analysis as we will need to identify the actual walking path of the tags.

The zone connectivity information consists of two elements: the door coordinates between two zones and the walking distance between them.

An example of the connectivity matrix is shown in Table 4.4. The matrix contains information on the coordinates of the connecting door and the walking distance in the format

Table 4.4: Example of connectivity matrix based on the example floor plan shown in Figure 4.1

	Ward	Exam room	Control room	Corridor
Ward	-			$(187, 320), 467$
Exam room		-	$(602, 258), 378$	$(340, 320), 340$
Control room		$(602, 258), 378$	-	$(825, 320), 546$
Corridor	$(187, 320), 467$	$(340, 320), 340$	$(825, 320), 546$	-

of $(X_door, Y_door), Dist_w$. The calculation of walking distance between two connected zones Z_1 and Z_2 is as follows:

$$Dist_w = Dist_{Z_1 \ to \ door} + Dist_{door \ to \ Z_2}$$

Where

$$Dist_{Z_1 \ to \ door} = \sqrt{(X_{Z_1} - X_{door})^2 + (Y_{Z_1} - Y_{door})^2}$$

and

$$Dist_{door \ to \ Z_2} = \sqrt{(X_{door} - X_{Z_2})^2 + (Y_{door} - Y_{Z_2})^2}$$

In rare cases, there can be more than one door connecting two zones directly. As IR based RTLS only provide zone level precision, it is impossible for the system to tell which particular door did a tag go through. Thus we pick the door which brings us the shortest walking distance between these two zones.

4.1.4 Tag Information

RTLS tag plays an important role in the system. It receives the infrared signal from the monitors and report its identity and the received infrared ID to the system. For the RTLS system adopted in this study, there are 3 types of tags, shown in Figure 4.2.

(a) Staff tag (b) Asset tag (c) Patient tag

Figure 4.2: Examples of different types of RTLS tags

The tags are in different shapes in order to be easily attached to different types of entities, despite the difference in shape, they have the same features.

Tag information consists of two parts: hardware configuration and tag assignment. As the hardware configuration is trivial for this study, only the tag assignment information will be introduced.

For privacy concerns, we do not track to a particular person in this study, i.e. we do not match a tag to a person. Instead, we only record the tag holder's role if the holder is a staff.

The tag assignment data format is simple, stated in Table 4.5.

The tag assignment data needs to be manually recorded.

Table 4.5: Properties of tag assignment data

Property	Comment
Effective date	The date from when is this assignment effective
Tag ID	The ID of the tag
Tag role	The assigned role of the RTLS tag. For instance, "nurse", "physician" or "technician". This information is retrived from the tag assignment
Tag type	The type of the tag. It can be "staff", "asset" or "patient"

4.2 RTLS Data Cleaning

In this section, we firstly introduce the causes of data quality problems in the RTLS data. Secondly, we introduce the strategy of cleaning the RTLS data. The strategies of cleaning the RTLS data are existing works, we introduce how we implement the strategies in our RTLS data cleaner.

4.2.1 Problems Causing Missing or Erroneous Data

Although the IR based RTLS we adopted in this study can achieve zone level precision, it has several shortcomings which may lead to missing or erroneous data.

Weak or blocked IR signal

Unlike radio signal, the infrared signal has shorter wavelength in the electromagnetic spectrum. Infrared signal has less penetration power. Although this less penetration power can avoid troubles of telling which side of the wall is a tag located at, it introduces difficulties for the tags to receive signals when the tag is blocked, for instance by a thick jacket, blanket, medical equipment etc.. In this case, the tag will report 0 as the monitor ID to the star. This is called "zero-location report"

Zero-location reports can also be caused by weak IR signals, for instance, when there is only one monitor installed in the middle of the ceiling of a big zone, the tag may not be able to pick up signal when it is located at the corners of the zone. We try to avoid this by installing multiple monitors in the same zone using monitor groups. In a monitor group, there is one master monitor and one or more slave monitors, the tag will always report the monitor ID of the master when the tag is in this zone, even if it picks up the IR signal from the slave monitor.

Missing reports

The tags are programmed to report their location in variate intervals, for instance, when they are "busy" - there are continuous motions in a certain period, they report themselves to the stars every 6 seconds. When there are no motions for a period of time, they will report themselves less frequently to reduce battery use.

However, there can be two types of missing reports under this mechanism: firstly, the tag may have two or more transitions in one interval. Suppose a tag reports itself in the exam room (see the example floor plan in Figure 4.1) at the first second, and then it moved to the corridor in the

third second and then entered the ward in the sixth second and stayed there. As the next location report will be made at the seventh second, this next location report will show that the tag is in the ward. In this way the tag did not get a chance to report its location when it was in the corridor, thus this report was missing.

Secondly, as the tag location report is based on radio, sometimes the radio signal can be weak and the report can be lost, especially when the distance between the tag and the star is too far or there are many walls between them.

IR signal interference

Another problem of the IR based RTLS is that the infrared light emitted by the monitors can bounce off walls or penetrate transparent glasses and possibly be picked up by a tag when the tag is located outside the zone but nearby the opening door or outside of the transparent glass window. This can get more tricky when both sides of the door have RTLS monitor installed. This can possibly lead to hopping in location reports. For example, using the floor plan in Figure 4.1, a tag is stationary outside of the opening ward door, it may report itself in the ward and then in the corridor alternatively.

4.2.2 Null Location Report Interpolation

Null location report interpolation is done to address the data quality problem caused by weak or blocked IR signal introduced in Section 4.2.1. When the location of a location report is missing, the following strategy is used to fill in the missing location.

For a continuous set of missing location reports $\{LOC(T_i, t_{mmin}) = \varnothing, LOC(T_i, t_{mmin+1}) = \varnothing, \cdots, LOC(T_i, t_{mmax}) = \varnothing\}$, we find the two nearest valid location reports: $LOC(T_i, t_a) = L_1$ and $LOC(T_i, t_b) = L_2$ where $a < mmin$ is the maximal timestamp when $LOC(T_i, t_a) \neq \varnothing$ before the missing location reports and $b > mmax$ is the minimal timestamp after the missing location reports when $LOC(T_i, t_b) \neq \varnothing$. If

$$L_1 = L_2,$$

then we fill the missing location reports with this location:

$$\{LOC(T_i, t_{mmin}) = L_1, LOC(T_i, t_{mmin+1}) = L_1, \cdots, LOC(T_i, t_{mmax}) = L_1\}$$

Otherwise, if their locations disagree, then we consider the missing location reports as outside of monitored area

$$\{LOC(T_i, t_{mmin}) = \varnothing, LOC(T_i, t_{mmin+1}) = \varnothing, \cdots, LOC(T_i, t_{mmax}) = \varnothing\}$$

In this case, we fill the missing location reports with a code which indicates “out of monitored area”.

4.2.3 Path Interpolation

Path interpolation is done to address the data quality problem caused by missing reports introduced in Section 4.2.1. The step consists of two parts: detection of missing location reports and interpolating location reports.

To detect missing location reports, we iterate the location reports and check every 2 adjacent location reports R_1, R_2 to investigate whether the zones reported in the two reports are connected. This is done using the connectivity matrix introduced in Section 4.1.3.

If there does not exist a path which connects the two zones directly, then there are missing report/reports between the two adjacent reports. To fill in the missing reports, we use Dijkstra's algorithm [33] to find the shortest path between R_1 and R_2.

Suppose the detected shortest path has n nodes or zones except the start and end nodes, the zones in the shortest path are $Ri_1, Ri_2, \ldots, Ri_n$ along the path. For any zone in the path, we create a mock up location report with this zone and add it between the two location reports R_1 and R_2. Thus the interpolated path is:

$$\{R_1, Ri_1, Ri_2, \ldots, Ri_n, R_2\}$$

4.2.4 Hopping Detection and Removal

Hopping detection and removal is used to address the data quality problem caused by IR signal interference introduced in Section 4.2.1.

To detect "hopping" reports, for each tag, starting from the first location record, we detect for every 3 adjacent location records R_1, R_2, R_3 whether they match the pattern

$$\{LOC(T_i, t_m) = L_1, LOC(T_i, t_{m+1}) = L_2, LOC(T_i, t_{m+2}) = L_1\}$$

where $L_1 \neq L_2$.

If such set of continuous location records series exists, then we check whether

$$t_{m+2} - t_m < k$$

where k is a parameter which indicates the time threshold for defining "hopping". The definition of k depends on how often the tags should report themself, i.e. how many signals it transmits every second. We note this transmition frequency as f. Typically k is equal or slightly greater than $\frac{2}{f}$.

If,

$$t_{m+2} - t_m \leq k,$$

then we decide that there is a hopping in $\{R_1, R_2, R_3\}$ and R_2 is considered as noise and thus will be removed, the next checking will be performed on $\{R_3, R_4, R_5\}$ as R_2 is removed. Otherwise, if

$$t_{m+2} - t_m > k,$$

then R_2 is considered correct and no change will be done to $\{R_1, R_2, R_3\}$, so the next checking will be performed on $\{R_2, R_3, R_4\}$.

4.3 Enriching the Processed RTLS Data

To make the processed data more readable and intuitive, we enrich the raw data according to the metadata. The important fields added during this process are listed in Table 4.6

Table 4.6: Added properties of in the enriched RTLS location log data

Property	Comment
Tag type	The type of the tag. It can be "staff", "asset" or "patient", this is derived from the tag ID and the tag metadata
Zone ID	The ID of the zone where the detected monitor is installed. This information is derived from the monitor ID to zone ID mapping in monitor metadata
Tag role	The role of the RTLS tag. For instance, "nurse", "physician" or "technician". This information is retrieved from the tag assignment
Zone name	The name of the zone, from zone information metadata
Zone type	The type of the zone, from zone information metadata
Zone center coordinates	The coordinates of the zone, from zone information metadata
From door coordinates	The coordinates of the door from which the tag entered the current zone, this value will be the same as zone center coordinates if the previous tag location reports the same zone, i.e. no transition happened since last report

Tag type and tag role are filled by looking up tag assignment metadata using tag ID. Zone ID is filled by looking up monitor information metadata using monitor ID. Zone name, zone type and zone center coordinates are generated by looking up zone information metadata using zone ID. From door coordinates are calculated using zone connectivity metadata when the zone ID of the previous location report is different from the current location report. Otherwise the from door coordinates are set to be the same as the zone center coordinates.

4.4 RTLS Data Zipping

Tags report their location periodically even when they are not moving. This generates large volume of duplicate RTLS data. We apply data zipping strategies to reduce the storage use and boost query speed.

In this study, zipping means to group adjacent similar location reports and to form report intervals instead of storing all the full reports. In our study, we consider adjacent similar location reports as location reports that are next to each other for a specific tag along the epoch time whose properties are identical except the epoch time. Suppose we have a series of location reports:

$$\{L_{a1}, L_{a2}, \ldots, L_{an}, L_{b1}\}$$

Where the properties of $L_{a1}, \ldots, L_{an}$ are identical except their time stamps and L_{b1} has different properties than $L_{a1}, \ldots, L_{an}$, then a zipped location report Lz_a is generated where the start

Table 4.7: Example of cleaned RTLS reports(the content is mocked up)

No.	Epoch time	Tag ID	Zone Name	Motion	Buttons	...
1	2017-01-01 08:00:00	TAG A	Corridor	yes	no	...
2	2017-01-01 08:00:06	TAG A	Corridor	yes	no	...
3	2017-01-01 08:00:12	TAG A	Control room	yes	no	...
4	2017-01-01 08:00:18	TAG A	Control room	no	no	...
5	2017-01-01 08:00:24	TAG A	Control room	no	no	...

Table 4.8: Example of zipped clean RTLS reports based on the location reports in Table 4.7

Start time	End time	Tag ID	Zone Name	Motion	Buttons	...
2017-01-01 08:00:00	2017-01-01 08:00:12	TAG A	Corridor	yes	no	...
2017-01-01 08:00:12	2017-01-01 08:00:18	TAG A	Control room	yes	no	...
2017-01-01 08:00:18	...	TAG A	Control room	no	no	...

time of Lz_a is the epoch time of L_{a1} and the end time of Lz_a is the epoch time of L_{b1}.

For instance, consider the location reports shown in Table 4.7:

Cleaned location reports 1 and 2 are similar, 4 and 5 are similar. The zipped RTLS reports based on them are shown in Table 4.8:

4.5 CRF Data Introduction

The Case Report Form (CRF) data is collected by observers in the hospital using an iPad app. CRF has a fixed structure, describing the start time of a fixed set of events in procedures.

4.5.1 CRF Data Structure

The CRF data consists of 2 parts. The first part is the case properties and the second part is the event times of the cases.

- Case properties
  - Case ID
  - Case Date
  - Procedure Type
  - Comments
- Event times
  - Patient registration time
  - Patient registration type
  - Patient arrive at waiting room time
  - Patient arrive at waiting room type

– ...

For each event, there is a type for the recorded event time. It indicates whether the event time is inputted when the event starts, filled later or missing.

4.5.2 Data Quality Problems

Missing Procedures

Some procedures may not be found in the CRF data because the recording of CRF data is not mandatory in the hospital's regulations. For instance, when there is an emergency exam happened at midnight when the observer is not at work, then this procedure will not be recorded although the RTLS data for this procedure is automatically collected.

Missing or Erroneous Event Time

Recording CRF data is only a part of the observer's work. It is possible that several events in a procedure are missing in the CRF data. It can also be the case that the observer inputted the same event time for multiple events.

4.6 CRF Data Cleaning

As introduced in the previous section, CRF data suffers from two categories of problems. We apply data cleaning methods to improve the quality of the CRF data. The goal of CRF cleaning is not to fill out the missing procedures or events, but to detect erroneous event times and filter them out to guarantee the accuracy of the event times.

Two strategies are used to clean the CRF data based on process mining models.

4.6.1 Process Model Conformance

The idea of this strategy is to first construct a "correct" process model and then apply conformance check between this model and CRF event logs and filter out the non conforming events.

The reason behind this strategy is: we suppose all the events should take place according to a process model. For instance, the "first incision" event cannot happen before the "patient is clear for procedure" event.

4.6.2 Outlier in Event Time Intervals

In this strategy, we check the time interval between two adjacent events. If the absolute difference between the time interval and the mean time interval of that two types of events is higher than a threshold, usually 2 or 3 times of the standard deviation, then the events are filtered out.

Chapter 5

Visual Analysis of RTLS and Workflow

In this chapter, visual analysis tasks on the workflow based on RTLS and CRF data is introduced. The introducing of RTLS technology in this study enables the hospital to not only know when did each event in workflows happens but also know staffs' locations when the events are taking place.

The tasks in this chapter address two of the business use cases: workflow playback and transition sequence analysis. These two tasks belong to the visualization task in the data analytics tasks.

5.1 Workflow Playback

The objective of this task is to correlate the RTLS data and the CRF data to intuitively show the correlation between staffs' movements and the ongoing workflow events.

5.1.1 Data for Workflow Playback Visualization

As stated above, workflow playback requires two sources of data. The RTLS data adopted is the cleaned and zipped RTLS data produced following the data cleaning steps introduced in Chapter 4.

5.1.2 Design and Implementation

User Interface Design

The workflow playback feature is a single page visualization with different modules.

Figure 5.1 depicts the layout design of the workflow playback page. The page consists of 4 parts.

Time axis Time axis is a control block where the playback time is displayed. It also allows user to drag to jump to a specific time point.

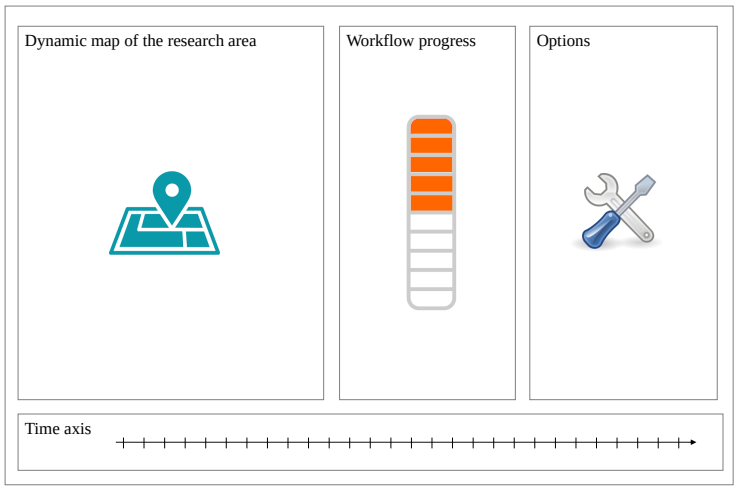


Figure 5.1: The layout design of the user interface

Dynamic map Dynamic map shows not only the floor plan of the research area, but also intuitively shows known tag locations of a specific time point. It also supports animation to demonstrate the transition path of the tags.

Workflow progress Workflow progress shows the progress of ongoing procedures given a time point. Different events of the workflow will be marked and shown according to the time when did each event happen.

Options Options enables the users to configure the visualization or load new location data in it. The configurations include tag filter where users can select particular tags according to keywords, a slider to change the playback speed and buttons to clear playback cache in the database and show/hide the time axis.

Implementation of Workflow Playback

The tool is built with browser / server (B/S) architecture. The visualization is implemented using D^3 [34]. The data comes from the relational database (RDB). The back end is implemented using Java Web. JSON is used to convey data from the back end to D^3.

The visualization for workflow progress is relatively easy as the data structure for workflow itself is simple, the related workflows in a playback can be pre-loaded into javascript. However, the difficult part is the map visualization. Because we not only show the current locations of the tags on the map, but also visualize the location change in animation when a transition is detected, i.e. for a tag, the location has changed since last report.

The strategy for the map visualization is: while the current playback time is flowing along the time axis, the browser continuously requests data from the back end using the current playback time. The tags are shown in the map according to the position they are at the given the specified time stamp.

The data structure for the tag location is listed in Table 5.1. When the front end submits a request for tag locations to the back end over Ajax, the back end server generates a JSON array

of tag locations and send back to the front end.

Every time when a new list of tag locations is requested, the D^3 script will check whether there has been a change for each tag location, including entering / updating /exiting. The logic for entering / updating tag location is depicted in algorithm 1.

for *tag loc* $\in$ *tag loc list* **do**
  if *is a new tag* **then**
    draw the tag on the map according to tag (X,Y);
    add the tag to known tags;
    add current coordinates to the tag's last known coordinates;
  else if *is an existing tag* **then**
    if *the location has changed* **then**
      initiate animation from the last known coordinates to the door coordinates;
      initiate animation from the door coordinates to the new coordinates;
    else
      continue;
    end
end

Algorithm 1: D^3 logic for displaying tag locations on the map

Exiting is not implemented because the back end server will always return the location of each tag. When a tag is not inside the monitored area, then the tag will be shown in a particular location in the map meaning they are temporarily out of the bounds. These tags will not be removed from the map.

5.1.3 Result

An example screen shot of the workflow playback tool is shown in Figure 5.2.

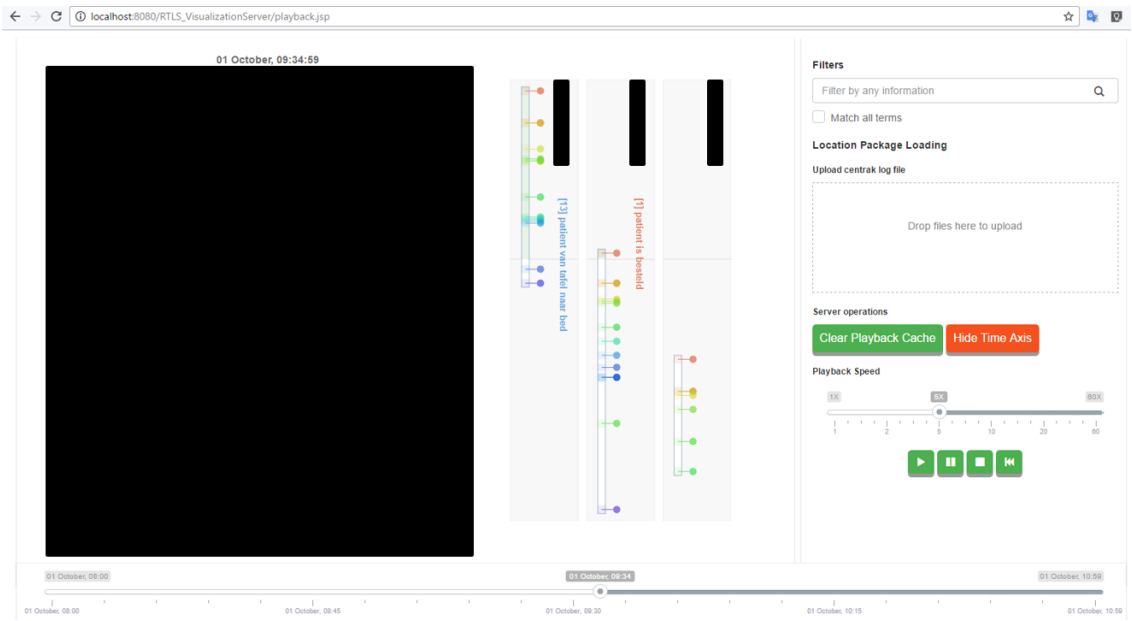


Figure 5.2: An example screen shot of the workflow playback tool

Table 5.1: Added properties of in the enriched RTLS location log data

Property	Comment
Epoch start time	The start time of this zipped tag location record. The Epoch start time is in UNIX timestamp format, i.e. the number of seconds since 1970-01-01
Epoch end time	The end time of this zipped tag location record. The Epoch start time is in UNIX timestamp format, i.e. the number of seconds since 1970-01-01
Tag ID	The ID of the RTLS tag, which is an integer value
Monitor ID	The ID of the monitor this RTLS tag detected. It is also an integer
Motion status	This is a boolean field. There are motion sensors equipped in Centrak RTLS tags. When the sensor detects a movement, this field will be set as *true* in the next location report.
Button status	including boolean values for each button. If a button is pressed, then the corresponding boolean value will be *true*
Battery level	which is a boolean value. This field will be *true* if the current tag has low battery
Tag type	The type of the tag. It can be "staff", "asset" or "patient", this is derived from the tag ID and the tag metadata
Zone ID	The ID of the zone where the detected monitor is installed. This information is derived from the monitor ID to zone ID mapping in monitor metadata
Tag role	The role of the RTLS tag. For instance, "nurse", "physician" or "technician". This information is retrieved from the tag assignment
Zone name	The name of the zone, from zone information metadata
Zone type	The type of the zone, from zone information metadata
Zone center coordinates	The coordinates of the zone, from zone information metadata
From door coordinates	The coordinates of the door from which the tag entered the current zone, this value will be the same as the zone center coordinates if the previous tag location reports the same zone, i.e. no transition happened since last report

5.2 Transition Sequence Visualization

In transition sequence visualization, we target to display an overview of the tags' movement sequence during procedures. The difference between this tool and the workflow playback tool is that this tool shows the transition patterns for multiple exams in the same plot while the playback tool shows the transitions for each individual exam. Also we do not use the floor plan in this visualization. A snapshot of the playback tool shows only the tag location status and procedure progresses at a given time point, however, in the transition sequence visualization, the visualization is static and shows the frequent transition patterns of a time window.

5.2.1 Visualization Framework

In this tool, parallel coordinates [35] is used. Parallel coordinates are frequently used for high-dimensional geometry and multivariate data.

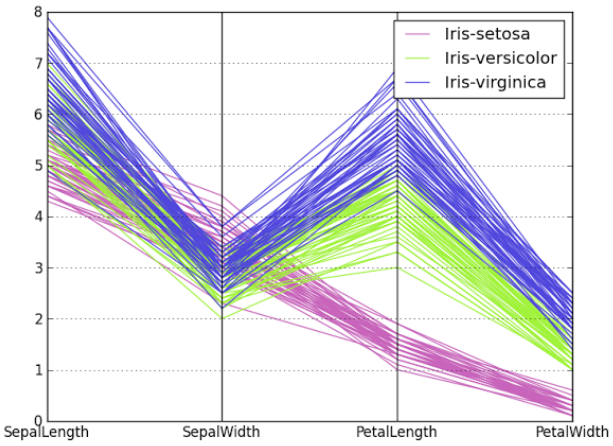


Figure 5.3: An example of using parallel coordinates visualization on Iris data set [1]

Figure 5.3 shows an example of parallel coordinates visualization using the Fisher's Iris data set [1]. In this visualization, the columns are different variables while each polyline represents a data point in the iris data set. Using this visualization, we can intuitively observe the correlations between different variables for different iris flower species.

In our visualization, the x-axis will be the different timestamps while the y-axis represents different zones or zone types.

5.2.2 Data Preparation

As we want to plot tags' transitions during exams, our study object here is a tag's transitions during an exam. Starting from the first second, we collect this tag's location every n seconds until the exam finish.

Besides the tag's location sequence, the tag's ID, role and the exam ID are added in the first three columns. This will enable us to easily filter the visualization according to tag ID, tag role or

Table 5.2: Mapping between zone types to zone type IDs codding

Zone type	Zone type ID
Out of reach and observation room	0
Corridor	1
Target control room	2
Target exam room	3
Storage room	4
Other exam & control rooms	5

according to particular exam cases. And it enables us display traces in different colors according to the tag type, assuming different roles have different transition patterns.

Data Alignment

Before preparing the transition sequence visualization, the target procedure time span needs to be chosen. For instance, we want to visualize the transitions from the time when a patient steps in the exam room (event E_a) to the time the patient leaves the exam room (event E_b), we will need to prepare the time stamp of the two events T_a and T_b for all the procedures first. Then, for each procedure, starting from the time T_a until T_b, we query the tag locations for each involved tag in the procedure and add them into the tags location sequence. A resolution parameter is set to decide the time interval between two location records in the location sequence. This process is described in pseudo code 2.

$seq[n_{tags}]$ = empty list ;
for ($loop\_time = T_a$; $loop\_time < T_b$; $loop\_time = loop\_time + resolution$) **do**
  for tag_t *in tags* **do**
    $loc_t = location\_query(tag_t, loop\_time)$;
    $seq[n_{tags}].append(loc_t)$
  end
end

Algorithm 2: Tags sequence data generation for a procedure

Zone Coding

In studies where many zones are defined, the visualization can become chaotic and not intuitive. To keep a clean visualization, in this case we use the type of the zone instead of the zone id as the y-axis value.

In this study, we use the coding mapping listed in Table 5.2.

5.2.3 Visualization Implementation

Similar to the playback tool, the transition sequence visualization tool is also implemented with browser / server architecture. We use D^3 to implement the browser end of this visualization tool. An existing sequence coordinates visualization package Parallel Coordinates [36] is adopted. The server end is implemented by Java Web. JSON is used to convey data from the server to the

browser.

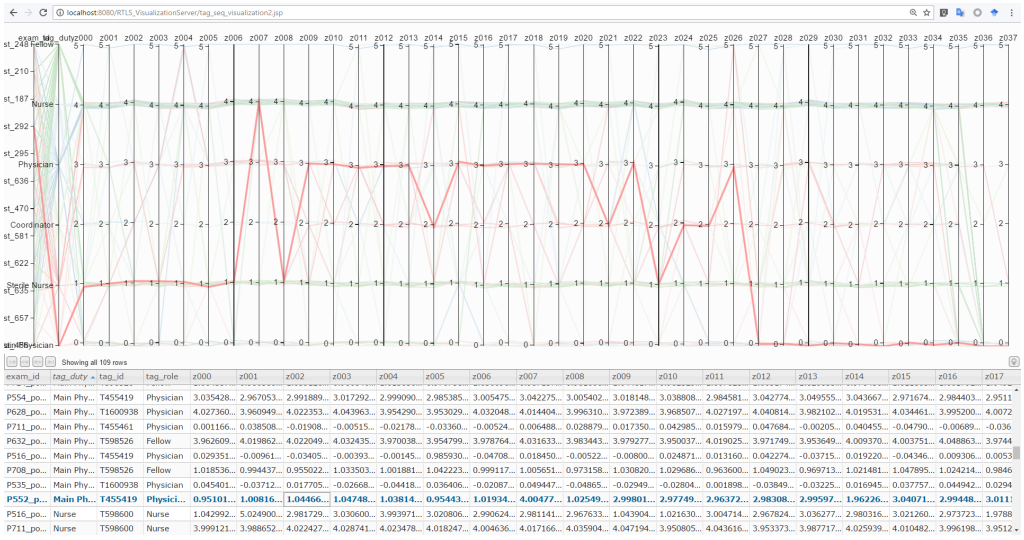


Figure 5.4: An example of the transition sequence visualization

An example of the transition sequence visualization is shown in Figure 5.4. This visualization depicts all tag transitions between "patient enters the exam room" and "patient leaves the exam room". The trace of a main physician in a procedure is highlighted in this diagram, we can observe that the physician was first in the corridor, then he went to the storage room and came back to the corridor. Later he entered the exam room and was walking back and forth between the exam room and the control room. Finally he exited the exam room and left the monitored area.

Data Jittering

To avoid overlapping of the same transition traces made by different tags or in different procedures, we apply data jittering to the zone type codes. By adopting this method, frequent transition paths will appear thicker in the visualization.

Chapter 6

Event Identification using Machine Learning

In this chapter, the problem of identifying workflow events is defined and the approach of building, training and evaluating a machine learning model based on the RTLS, CRF and machine log data to solve this problem, as well as the approach to deploying this machine learning model onto a real-time setting to provide online automated event identification are introduced.

6.1 Problem Formulation

The objective for event identification is to use the RTLS data to automatically identify the workflow events in procedures. There are different types of workflow events in a procedure. For instance, in our study, there are 15 workflow events listed in Table 6.1.

To simplify this problem, we convert the problem from "What workflow event is ongoing at the moment?" to "Whether workflow event e is ongoing at the moment?". In other words, we convert the multiclass classification problem to a binary classification problem.

There are several purposes for this simplification. Firstly, in the cases where different types of events are not mutually exclusive, we cannot apply a multiclass classification algorithm as we may need to give more than one class labels for each data instance. A combination of multiple binary classification algorithms can solve this problem. Secondly, we may not be interested in predicting all events, especially in workflows which consist of tens or hundreds of events. It is difficult to model and evaluate multiclass classification models when the number of classes are large. Lastly, as we want to tackle this problem by attempting various machine learning algorithms, this simplification would enable us to use machine learning algorithms that are not naturally suitable for multiclass classification problem for instance SVM and logistic regression.

Thus, the problem can be formulated as: given the location information of all the tags loc_{tags}, predict a label about whether an event s is *ongoing* or *not ongoing*.

Table 6.1: List of events in our CRF dataset

Event number	Event name
1	Patient is registered
2	Patient arrived in waiting room
3	Patient arrived in exam room
4	Patient laid on table
5	Physician is called
6	Physician arrives in the exam room
7	Patient is clear for procedure
8	Start incision
9	Close incision point
10	Called to retrieve the patient
11	Physician leaves the exam room
12	Physician leaves the control room
13	Patient retrieved from table to bed
14	Patient dismissed
15	End of research

6.2 Feature Preparation

As discussed in Section 3.4.2, it is not feasible to use the cleaned RTLS data directly for training the prediction model. In the beginning of this section, we discuss why merely the cleaned RTLS does not provide us enough information to predict a workflow event.

Table 6.2 shows an example fragment of the cleaned RTLS data. From the data structure, we can observe that although the columns are structured, the rows can be in different orders and there are dependencies between data from different rows. For instance, the location of "TAG A" at "2017-01-01 08:00:06" highly depends on the location of the same tag at "2017-01-01 08:00:00". In addition, not all tags report their location in the same interval, in this case 6 seconds. For instance "TAG B" did not report its location at "2017-01-01 08:00:06" is possibly lost. It can also be the case that the tag was in sleeping mode due to inactivity and increased its report interval.

One row of the cleaned RTLS data only tells us about the location of a tag at a time point,

Table 6.2: Example of the data structure of the RTLS data (the content is mocked up)

Epoch time	Tag ID	Zone Name	Other features
2017-01-01 08:00:00	TAG A	Corridor	...
2017-01-01 08:00:00	TAG B	Exam room	...
2017-01-01 08:00:00	TAG C	Control room	...
2017-01-01 08:00:06	TAG C	Control room	...
2017-01-01 08:00:06	TAG A	Control room	...
2017-01-01 08:00:12	TAG B	Exam room	...

Table 6.3: Example of the aggregating transient location status by zone

Ward	Exam room	Control room	Corridor
n1	p1, n2, n3	f1	p2

Table 6.4: Example of transient location status aggregated by tag

n1	n2	n3	p1	p2	f1
Ward	Exam room	Exam room	Exam room	Corridor	Control room

however it does not contain a macro information about the overall tracked entities. For example, to predict at a time point whether there is a workflow event ongoing, we would like to aggregate and use the status of all the tags instead of making the prediction based on merely the transient location of a single tag.

6.2.1 Aggregated Transient Location Status

The purpose of aggregating location reports is to form a data point which can represent the transient location status of all the tags tracked in the RTLS system. There are two ways to aggregate: by zones and by tags.

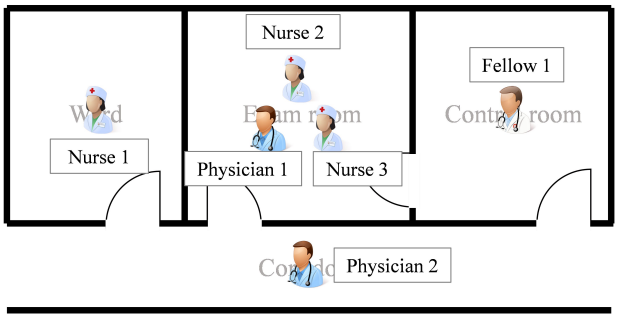


Figure 6.1: An example of tags' locations in the example floor plan

Figure 6.1 shows an example of tags' locations in the example floor plan. In this example, there are 6 tags in 3 different roles, distributed in 4 zones. To represent the location status of all the tags, we can either break down the location status by zones so that we describe the tags in each zone, alternatively we can break down the by tags so that we describe the location of each tag. Two examples are given in Table 6.3 and Table 6.4. Here we use abbreviation n as nurse, p as physician and f as fellow.

Table 6.3 gives an example of representing transient location information by aggregating by zones. In this approach, we have 4 columns representing the 4 monitored zones. For each column, tags in the zone represented by this column are listed.

On the contrary, we can represent the transient location information by aggregating by tags. An example is shown in Table 6.4.

Practically it is advisable to aggregate by zones instead of by tags because in the real world

Table 6.5: Example of a transient location status aggregated by zone and tag role

Ward			**Exam room**			**Control room**			**Corridor**		
Nurse	Phy.	Fellow	Nurse	Phy.	Fellow	Nurse	Phy.	Fellow	Nurse	Phy.	Fellow
1	0	0	2	1	0	0	0	1	0	1	0

settings, zones settings are relatively static. It is uncommon to add/remove or change the floor plan. In contrast, it is common for hospitals to add new tags or replace old tags with new tags. Especially there are RTLS tags designed for one time use for the patients. Another reason for not advising aggregation by tag is, the number of tags in the system is usually much higher than the number of zones. A big number of columns will increase the model complexity and may lead to "the curse of dimensionality".

However, in the example shown in Table 6.3, the data structure is more complex than aggregating by tags because the values for each column is a list of tag IDs. We address this problem by further aggregating the data by roles and only keep the numbers of tags of each role of each zone.

Table 6.5 shows an example of the updated data structure. This data structure is relatively stable. Firstly, the floor plan and staff role classifications of a hospital do not change often. Secondly, the data structure will remain the same when new tags are added, existing tags are replaced or removed. An extra benefit of this data structure is that the values are the counts of the tags. This avoids the privacy concerns because it will be not possible to track individual tags from this aggregated data.

6.2.2 Single Point Vs. Sequence Prediction

The aggregated data in Section 6.2.1 gives only the system state at a single time point. While we can train machine learning models on this data, we also explore the possibility of making predictions (including classification) based on a sequence of system states. For instance, we can make predictions based on the sequence of transitions made by the tags in a 10 minutes window.

The reason is, a sequence of tag locations over time gives insights on the trend of the system. This trend could provide powerful features for predicting workflow events. For instance, when a nurse tag leaves the ward together with a patient tag to the corridor followed by entering the exam room, the chances that this patient is going to be examined is high. On the contrary, if we only know the patient and the nurse's location in a single time point we lack the context which contains information on the what has happened in the past.

Table 6.6 depicts an example of an aggregated location status sequence by zone and tag role for a 1 minute time window. In this sequence, there are 6 transient location status records. The time interval is constantly 10 seconds.

The data structure of the location sequence is similar with the transient location status. The difference is that one sequence consists of a fixed number of transient location status items and the items are ordered by time and have a fixed time difference between each other.

In Section 6.3, when we train the sequence mining model, we feed the each sequence to the model in the form of a matrix. The entire matrix, for example from Table 6.6, will be considered as one training sample instead of six.

Table 6.6: Example of a location status sequence aggregated by zone and tag role for a 1 minute time window

Seq. No.	Ward			Exam room			Control room			Corridor		
	Nurse	Phy.	Fellow	Nurse	Phy.	Fellow	Nurse	Phy.	Fellow	Nurse	Phy.	Fellow
1	1	0	0	2	1	0	0	0	1	0	1	0
2	1	0	0	2	0	0	0	1	1	0	1	0
3	0	0	0	2	1	0	0	1	1	1	0	0
4	0	0	0	2	1	0	0	1	1	1	0	0
5	0	0	0	3	2	0	0	0	1	0	0	0
6	0	0	0	3	2	0	0	0	1	0	0	0

Table 6.7: Example of sequential RTLS data with motion information

Seq. No.	Ward			...	Corridor			Motion		
	Nurse	Phy.	Fellow	...	Nurse	Phy.	Fellow	Nurse	Phy.	Fellow
1	1	0	0	...	2	1	0	4	2	1
2	1	0	0	...	2	1	0	5	3	0
3	0	0	0	...	2	0	0	2	4	1
4	0	0	0	...	2	0	0	3	1	1
5	0	0	0	...	3	0	0	3	2	2
6	0	0	0	...	3	0	0	4	1	1

6.2.3 Adding Tag Motion Information

As introduced in Section 4.1, the RTLS data contains not only the location information, but also the motion information detected by tags' motion sensors. We include the motion information as addition features to the existing location features.

As in Section 6.2.1 we choose to aggregate location information by zones and roles, we also aggregate the motion information of roles. Suppose we have r roles, r columns will be added to the data structure to indicate for each role the number of motions detected since the last time stamp.

Motion information will be added to both transient and sequential data. Table 6.7 shows an example of sequential RTLS data with motion information.

6.2.4 Adding Hour of the Day

In addition to the movement and motion data, we also add the hour of the day as an extra feature. This feature is normalized to a float value between 0 and 1. It is calculated by $hour(epoch\_time)/24$.

The numbers in the motion columns do not mean the number of tags which had motion since the last time stamp because a single tag can report multiple motions.

Table 6.8: Example of constructing the tag location buffer

Streaming RTLS location reports			Tag location buffer		
Epoch time	**Tag ID**	**Zone name**	**TAG A**	**TAG B**	**TAG C**
2017-01-01 08:00:00	TAG A	Corridor	Corridor	-	-
2017-01-01 08:00:00	TAG B	Exam room	Corridor	Exam room	-
2017-01-01 08:00:00	TAG C	Control room	Corridor	Exam room	Control room
2017-01-01 08:00:06	TAG C	Control room	Corridor	Exam room	Control room
2017-01-01 08:00:06	TAG A	Control room	Control room	Exam room	Control room
2017-01-01 08:00:12	TAG B	Exam room	Control room	Exam room	Control room

6.2.5 Time Alignment

Cleaned RTLS data is a type of streaming data. Any tag can report themselves to the system at any time. However, we want to get the location status of the entire system given a time stamp. Thus time alignment is required. The strategy adopted in this study to align the time and produce training data is: firstly define a main time line, it starts with the time when we get the first location report in RTLS and ends with the time when we get the last location report. As the collection of the RTLS data is continuous during the data collection period, we always can get the updated tag location given any time along the main time line.

To get the most recent tag location given any time, we firstly construct a tag location buffer. This buffer stores the most recent location of each tag. To construct this, we firstly sort all location reports according to the epoch time and then continuously feed the tag locations into this buffer. Table 6.8 demonstrates the process of updating the tag location buffer.

Using this tag location buffer, we can then construct the aggregated data along the main time line. From the start time of the time line, we advance it every r seconds suppose r is the resolution in time and take all the tag locations in the buffer to construct the transient location data for this time.

6.3 Model Training

6.3.1 Event Class Labeling

As formulated in Section 6.1, the problem of predicting workflow events from RTLS data can be converted to multiple binary classification problems for identifying each event. Thus we have multiple labels for each data point for different events.

Suppose we aim to identify i events, we produce i event labels for each data point. In the training process, we train i binary classification models for each of the i labels. An event label can either be 0, which represents the event is not ongoing or 1, which represents the event is ongoing.

Table 6.9 shows an example of created event labels for the exams. Suppose we aim to identify 3 events: “Exam preparation”, “Exam” and “Wrap up”, we create 3 labels respectively. For data point 1 and 2, event “Exam preparation” was ongoing, for data point 5, no event from the 3 target events was ongoing.

The event labels for one data point does not need to be mutually exclusive. This is because

Table 6.9: Example of created event labels

ID	RTLS features	Exam preparation	Exam	Wrap up
1	...	1	0	0
2	...	1	0	0
3	...	0	1	0
4	...	0	0	1
5	...	0	0	0

A workflow event is an continuous action performed by multiple people. It has a start time and an end time.

Transient Location Data Labeling

The transient location data describes the location status for tags for given time points. To assign a label regarding to event e to a transient location data item, we take the time stamp of the transient location data item and look up CRF data in the database to find:

$$\{x|T_{start}(x) \leqslant T(l) < T_{end}(x), type(x) = e\}$$

Where $T_{start}(x)$ and $T_{end}(x)$ are the start and end time of event x, $type(x)$ stands for the type of the event.

If,

$$\{x|T_{start}(x) \leqslant T(l) < T_{end}(x), type(x) = e\} = \emptyset,$$

which means no events are found given the criteria, then we set the event label of e as 0, otherwise the label will be set to 1.

Sequential Location Data Labeling

Labeling a sequence is more complex because a sequence is about a time window. Different from transient location data which can only be in the target event or outside of the target event, the time window of a sequence can be partly overlapping with the event.

Figure 6.2 listed 5 possible relations between a location sequence time window and an event time span. For relation a and b it is obvious that the event label is 1 and while for relation c the label is 0. In cases d and e where an event took place only in part of the time window, a threshold t is set for deciding whether the label should be 1 or 0. The calculation in this case is:

$$L_e = \begin{cases} 1 & \text{if } Overlap(s,x) \geqslant t \\ Discard & \text{if } 0 < Overlap(s,x) < t \\ 0 & \text{if } Overlap(s,x) = 0 \end{cases} \tag{6.1}$$

In the case where the overlap is less than t but greater than 0, then this window is discarded.

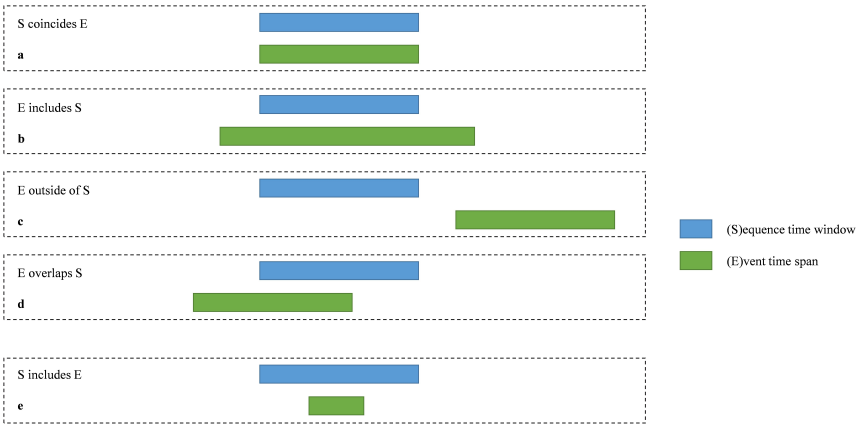


Figure 6.2: Possible relations between a location sequence time window and an event time span

6.3.2 Algorithm Selection

As stated in Section 6.1, the event identification task in this study is converted to multiple binary classification problems. Thus we choose binary classification algorithms to address this problem. We carry out experiments for both single point and sequential data using different algorithms.

Most conventional machine learning classification algorithms do single point classification: making class label predictions based on isolated data points without considering the correlation between different points. The single point classification algorithms are listed below:

- Logistic regression
- Decision tree. CART tree is adopted in this study
- Random forest
- Gaussian naive Bayes
- Neural network. Multilayer perceptron (MLP) is adopted in this study
- Support vector machine (SVM)

For predicting event labels using sequential RTLS data, long short term memory (LSTM) network is used.

6.4 Experiments and Results

This section describes the objectives, setups and results of the experiments related to the data analytics task of workflow event identification introduced in this chapter.

6.4.1 Experiment Objectives

There are three objectives that we aim to achieve in this experiment:

Table 6.10: Introduction of roles involved in the study

Role name	Description	Count
Physician	A physician the person who is in charge of the operation	4
Fellow	A fellow can be considered as a junior physician, they can be in charge of the operation if a physician is not present	3
Nurse	Nurses provide help to physicians or fellows in exams, provide care to the patients and do administration related works	6
Circulating nurse	Circulating nurses make preparations for the exams, monitor the patient and staff status and make records of the procedures, equipment etc.	3

- Verify that the concept and methodology of using RTLS data to automatically identify workflow events are correct and feasible.
- Compare the performance between models trained on single point data and models trained on sequential data to see whether the information about the movement sequences can improve the performance of the prediction.
- Investigate how close the performance of our event identification models is compared with the "gold standard" - the machine log data.

6.4.2 Data Set Specification

We have collected data from a radiology department of a hospital in the Netherlands. The data collection was done in 2 periods: before and after installing the next generation of interventional X-Ray (iXR) machine. For the event identification experiment, we use the data from the first period while in the experiment for the remaining time prediction we mainly focus on only the second period when the new iXR machine is in use. The first period lasted for 6.5 months and the second period lasted for 5 months.

RTLS Data

13 RTLS tags are monitored continuously during the first period and 16 RTLS tags are monitored continuously during the second period. The tags are divided into 4 roles. The roles are described in Table 6.10. In the first period, there are 3 of the 4 roles present: physician, fellow and nurse. All the 4 roles are present in the second period.

CRF Data

In parallel with the collection of RTLS data, CRF data is manually produced in the data collection phase. Dedicated observers are sent to the hospital to manually record the CRF data. 276 exam procedures which took place in the target exam room are observed and recorded.

Although there are 15 workflow events in the CRF data. These 15 events are not time lasting events, meaning that they are the only recorded with one time stamp when the events finish. As introduced in Chapter 6 and 7, the events we want to identify and predict need to be time lasting events that have start and end time stamp. Thus we choose two events in CRF data to form one time lasting event. In the experiment, we choose the "first incision" to "close incision point" events from CRF data and form a new event "exam" as the target event. The start time of the "exam" event is the time stamp of "first incision" while the end time of the "exam" event is the time stamp of "close incision point". The reason for choosing this event is that, this event is more difficult to identify than the other events, for example "physician enters exam room" event. Instead, the exam event is more complex and requires staffs with different roles to perform in collaboration.

The CRF data is comprehensive and relatively accurate. However, as stated in Section 4.5.2, it suffers from the problem that not all the workflow events are recorded in the CRF log.

Machine Log Data

The disadvantage of CRF data introduced in 6.4.2 causes difficulty in training and evaluating event identification models because the labels can be erroneous. For instance, the label will be 0 for the exam event when it is supposed to be 1 when the situation in the previous chapter happens. Thus, for the event identification experiment, we derive the exam event from the machine log instead.

We have access to the interventional X-Ray machine log database. The database stores all the event logs collected when the machines were used, including UI interaction, geometry movements, imaging etc., in this study we only use the imaging part of the machine log which includes interventional X-Ray exam acquisitions.

The exam start / end date time for all the exams done with the iXR machine in the target exam room happened in the first period of RTLS data collection are used. There were 660 exams matching this criteria.

Feature Preparation Strategy

This section introduces the approaches of preparing the features for the workflow event identification task described in Section 6.2.

Point Location Data Preparation Along the main time line we use a time resolution of 5 seconds, which means we take a snapshot of transient location states every 5 seconds. Using this strategy, we prepared 3,352,320 transient data items for the first period and 1,952,640 transient data items for the second period.

Sequential Location Data Preparation To form sequential location data, the length of the time window and the sliding window interval need to be set. The sliding window interval specifies the time difference between two adjacent windows. The two parameters are demonstrated in Figure 6.3.

In this study, we set the sliding interval to 1 minute / 60 seconds and we attempt different values of window lengths.

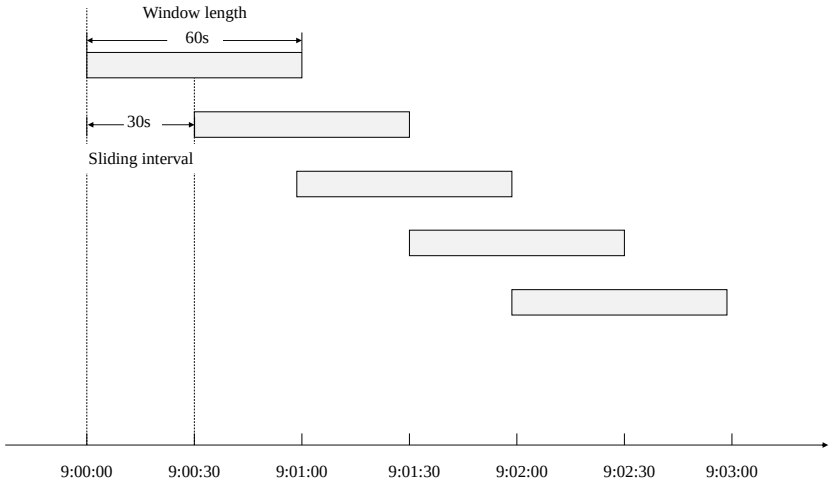


Figure 6.3: Forming sequential data

Table 6.11: Distribution of positive and negative samples in the exam identification data set

Data set	Size	Num. pos.	Num. neg.	%Pos.
Single point data	80,067	30,557	49,510	38.16%
Sequential data (5 minutes window)	74,389	29,513	44,876	39.67%
Sequential data (10 minutes window)	77,801	30,224	47,577	38.85%
Sequential data (15 minutes window)	80,067	30,557	49,510	38.16%
Sequential data (20 minutes window)	81,972	30,876	51,096	37.67%
Sequential data (25 minutes window)	83,451	31,068	52,383	37.23%
Sequential data (30 minutes window)	89,793	31,990	57,803	35.63%

6.4.3 Experiment Setup

This section introduces the setup of this experiment, including an introduction to the experiment data set, data set splitting and evaluation metrics.

Experiment Data Set

The event label of the exam identification data set is imbalanced. This is because the RTLS system is continuously collecting tag movement data even in non-working hours or weekends. We have done statistics of total exam time versus total monitoring time. The total exam time during the first period is 742 hours while the total monitoring time is 4,729. Thus the chance of having an exam at a random time point is lower than the chance that there is no exam going on.

We have removed duplicate data items especially in non-working hours and weekends when there is no tag movement at all for several hours. However, the label is still imbalanced.

Table 6.11 depicts the data sizes of the data sets for different type of models and distributions of positive and negative samples.

Training, Test, Validation Splitting

As discussed in Section 6.4.2, RTLS data items can be correlated, for instance, the transient RTLS data at 9:00:00 am can be highly correlated with RTLS data at 9:00:05 as they are near to each other in time and are likely to be in the same event, if there are events going on at this period. Thus we cannot split the data randomly because we do not want correlated data items to be split into different sets which may lead to falsely high performance.

The strategy used in our experiment is to split the sets according to time. So that RTLS data items from the same short time ranges will be always allocated into the same set.

The proportion of splitting is set to 0.7:0.15:0.15 for training test and validation.

Evaluation Metrics

Although simple metrics for instance accuracy, precision, recall etc. are of the most intuitive metrics that can be used in our event identification task, we have to be careful with using them because the samples in our data set is skewed. It is important to rely more on evaluation metrics that are not sensitive to label skewness and use accuracy etc. as references.

Receiver Operating Characteristics (ROC) Curve The Receiver Operating Characteristics (ROC) Curve is a type of plot which diagnosis the performance of a binary classifier when the classification threshold changes. The ROC plot has two axis: the false-positive rate or fall-out axis and the true-positive rate or sensitivity / recall axis.

By shifting the classification threshold, we get points in the ROC plot regarding to false-positive rate (fpr) and true-positive rate (tpr), by connecting these points we get a curve. When the points are away from the diagonal line to the upper part of the plot, they represent good classification performance. In contrast, when the points are away from the diagonal line to the bottom part of the plot, they represent poor classification performance. If the curve is very near to the diagonal line, then the classifier produces nearly no predictive power.

Area Under Curve (AUC) Based on the ROC curve, the Area Under Curve (AUC) metric represents the area below the ROC curve, it can be considered as the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example [37], i.e.

$$P(score(x^+) > score(x^-))$$

The AUC metric lies between 0.5 and 1.0. The more the AUC close to 1.0, the better the classifier performs. AUC is suitable to measure binary classifier's performance when the label of the data set is imbalanced.

Cohen's Kappa Cohen's kappa measures the agreement between two raters who each classify N items into C mutually exclusive categories [38]. It is computed as

$$k = \frac{p_0 - p_e}{1 - p_e} = 1 - \frac{1 - p_0}{1 - p_e},$$

where p_0 is the relative observed agreement or the accuracy and p_e is the expected agreement.

Table 6.12: Comparison of performances between different models

Model	Type	AUC	Kappa	Accuracy	F1
DecisionTree	single point	0.584	0.203	0.656	0.455
Naive Bayes	single point	0.671	0.263	0.708	0.451
LogisticRegression	single point	0.757	0.265	0.718	0.432
MLP	single point	0.703	0.314	0.710	0.521
RandomForest	single point	0.703	0.275	0.700	0.482
LSTM	single point	0.796	0.488	0.776	0.646
LSTM30m	sequential	**0.870**	**0.632**	**0.835**	**0.757**

The advantage of Kappa is that it is also suitable for data with unbalanced labels. This is especially useful in our experiment. The drawback of Kappa is that it is difficult to interpret the indices of agreement. Cohen's Kappa metric lies between 0 and 1. The more the value is close to 1, the better the classifier performs.

The calculation for other metrics: F1 score and accuracy are fundamental and not introduced in this thesis.

Accuracy can be regarded as the closeness to the baseline as the label of our data set is generated from the machine log data, which we consider as the baseline data. Accuracy lies between 0 and 1. The more this value is close to 1, the closer the model is to the baseline.

6.4.4 Evaluation Results

In this section, evaluation results are showed and compared for the workflow event identification task.

Results Overview

Table 6.12 shows the performances of all the models. For the sequential LSTM model, we have prepared sequential data set with window length 5, 10, 15, 20, 25 and 30 minutes. We have selected the model with the best AUC as the representative of sequential LSTM model. The sequential LSTM model achieves the best performance compared with the other models.

Table 6.13 lists the performances of LSTM models built on sequential data with various windows lengths. The LSTM model trained on the sequential data with 30 minutes window yields the best performance.

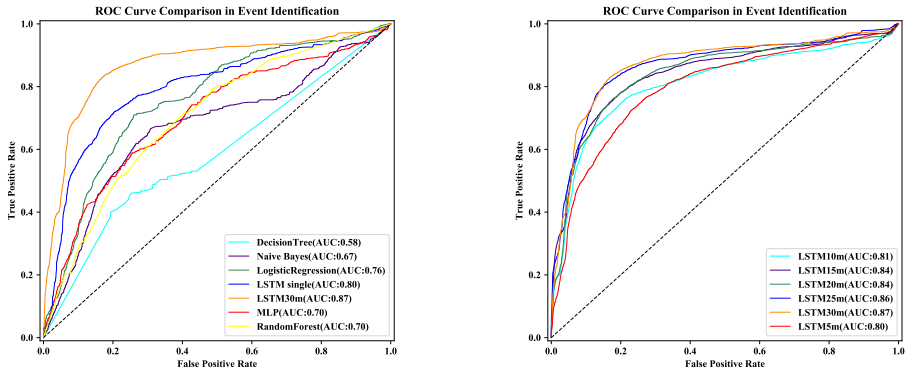
ROC Comparison

Figure 6.4a compare the ROC curves between different models. From this figure, it is obvious that the sequential data based LSTM model performs significantly better than LSTM and the conventional models based on single point data. Among the conventional algorithms, logistic regression yields the best ROC curve.

Figure 6.4b compare the ROC curves between LSTM models built on sequential data of different window lengths. From this figure, we observe that the classification performance of LSTM models

Table 6.13: Comparison of performances between LSTM with different window lengths

Model	Window length	AUC	Kappa	Accuracy	F1
LSTM	5 minutes	0.797	0.473	0.757	0.660
LSTM	10 minutes	0.810	0.546	0.799	0.692
LSTM	15 minutes	0.841	0.579	0.812	0.720
LSTM	20 minutes	0.840	0.554	0.805	0.697
LSTM	25 minutes	0.865	0.626	0.833	0.753
LSTM	30 minutes	**0.870**	**0.632**	**0.835**	**0.757**



(a) ROC curve comparison for different models (b) ROC curve comparison for LSTM built on sequential data with different window lengths

Figure 6.4: Comparison of ROC curves

with 25 and 30 minutes windows are similar and better than the other models. LSTM model with 5 minutes window yields the worst result among all the LSTM models. Generally, when the window length increases, the model achieves higher AUC.

6.5 Real-time Event Identification

One advantage of RTLS is that it can provide raw streaming location data in real-time. We have developed an online RTLS data cleaner based on HBase which can produce a cleaned RTLS location report in real-time after receiving a raw RTLS location report from the RTLS server. The cleaned data then is sent to the online feature processing server which generates transient or sequential location data periodically. The generated data is then sent to the online event identification server in which the trained model for event identification is deployed. The identified events will then be sent to the applications for different uses.

Figure 6.5 depicts the pipeline of real-time event identification.

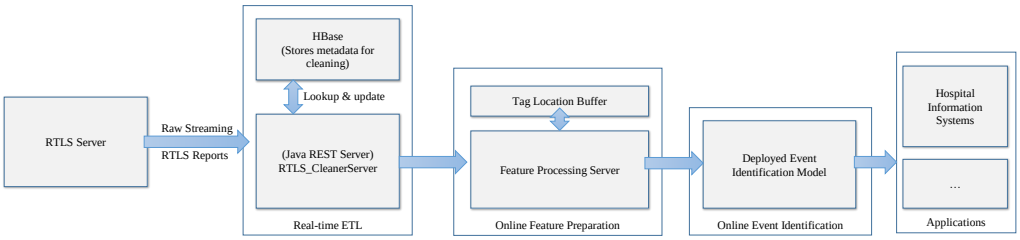


Figure 6.5: The pipeline of real-time event identification

Chapter 7

Event Remaining Time Prediction using Machine Learning

In this chapter, the problem of predicting the remaining times of ongoing workflow events is defined in the first section. The other sections introduce the approaches to build a machine learning model based on the existing RTLS and CRF data and to apply this model in a real world clinical settings.

7.1 Problem Formulation

Event remaining time prediction task is an extension of the event identification task depicted in Chapter 6. In this task, instead of identifying what workflow event is going on, we suppose a specific type of workflow event is already ongoing and the goal is to predict how much time is remaining before this event ends.

Two parts of input data are required. In addition to the RTLS data, which is also needed for identifying ongoing workflow event, the information for the ongoing event itself is also required. This data will tell the model what type of event is it trying to predict and for how long has it been ongoing.

Similar to the approach that we convert the multiclass classification problem to multiple binary classification problems in Chapter 6, we also build the prediction model for each of the event types that we want to predict.

The target of the prediction is the remaining time for the ongoing event before it ends. It can either be a continuous number of minutes / seconds or a time range, for instance 10 20 minutes. In this study, we give continuous values as the estimations of events remaining time.

7.2 Feature Preparation

The features used in event remaining time prediction is nearly the same as the event identification. However, as stated in Section 7.1, additional information about the ongoing event is also required. Also, we add a filter to keep only RTLS data that are related to ongoing events.

7.2.1 Event Related RTLS Data Filtering

As we aim to predict the remaining time of ongoing events, we do not make a prediction when there are no target events ongoing.

Filtering Single Point RTLS Data

Filtering single point RTLS data is simple as the RTLS data item has a single time stamp. We query this time stamp in the CRF events database and check at this time stamp whether an event with type e is ongoing. If not, the RTLS data at this time stamp is discarded.

Filtering Sequential RTLS Data

Filtering sequential RTLS data can be more tricky than filtering single point RTLS data as sequential RTLS data contains location information in a fixed length time window.

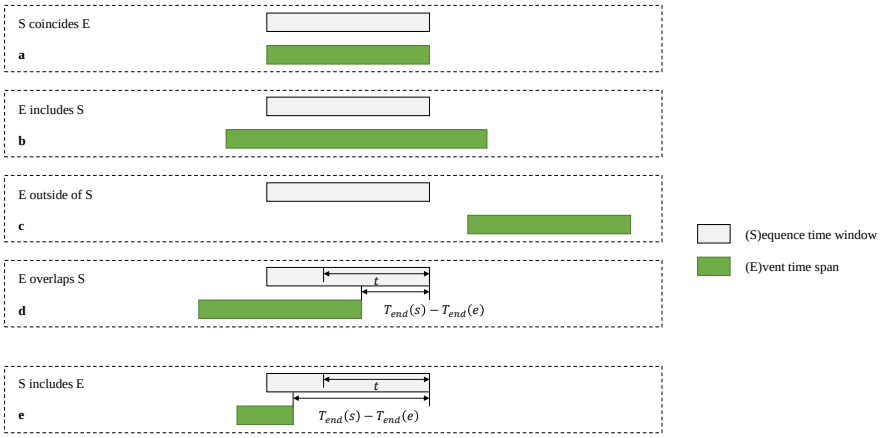


Figure 7.1: Filtering sequential RTLS data according to event time span

Figure 7.1 listed 5 possible relations between RTLS data sequence and target workflow events. For case a and b, we keep the RTLS data sequences because at the end of the sequence, an event has just ended or still ongoing. For case c, it is obvious that we discard this sequence, as it does not have relation with any events regarding to time. Case d and e are tricky because they both have overlapping with events. In these cases, we set a threshold t and check whether the difference between the end time of the sequence's time window $T_{end}(s)$ and the event's end time $T_{end}(e)$ is equal or greater than t.

If $T_{end}(e) - T_{end}(s) < t$ then this RTLS sequence is discarded.

7.2.2 Adding Event Elapsed Time

In addition to the location and motion aspects, we also add the elapsed time of the ongoing event as one extra feature.

Adding Elapsed Time for Single Point Data

Calculating the elapsed for single point data is relatively straightforward as the single point data represents the transient status at a specific time stamp. We take the time stamp of the transient location data item and look up CRF data in the database to find:

$$\{x | T_{start}(x) \leqslant T(l) < T_{end}(x), type(x) = e\}$$

As in the filtering process introduced in 7.2.1, we already filtered out the single point data items which do not have relationships with ongoing events and according to Section 6.1 there can be only one event of a specific type going on at the same time point, thus this query will always return one and only one event.

Suppose the event we have found is x, the calculation of event elapsed time of the single point data is depicted in equation 7.1.

$$elapsed\ time_l = T_{end}(l) - T_{start}(x) \tag{7.1}$$

Adding Elapsed Time for Sequential Data

For defining target for sequential data, as introduced Section 7.2.1, there can be different relations between the sequential data item and ongoing events. According to the situations listed in Figure 7.1, we list the method of calculating elapsed event time for each of the cases.

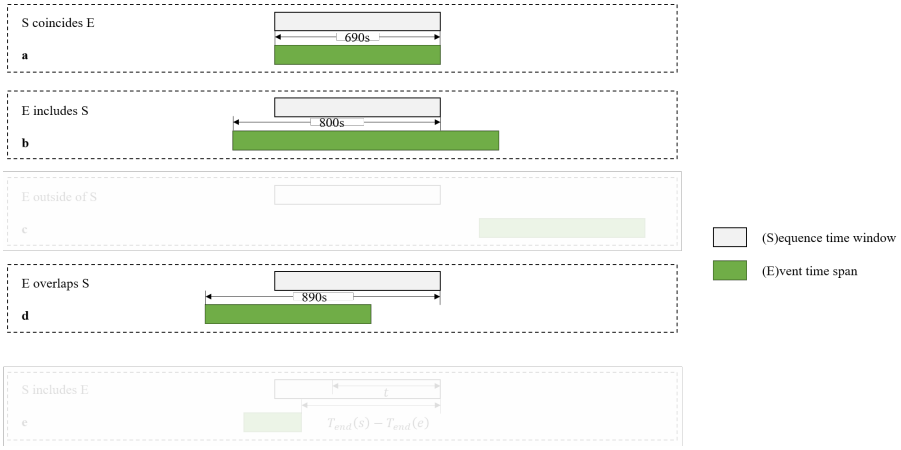


Figure 7.2: Calculating elapsed event time for sequential RTLS data

According to Figure 7.2, we do not need to take case c and e into consideration because they are already filtered out in the filtering process introduced in 7.2.1. For the other cases, suppose the matching event is x we calculate using equation 7.2

$$elapsed\ time_l = T_{end}(l) - T_{start}(x) \tag{7.2}$$

In addition, different from defining target for single point data, a sequential data item can possibly relate to multiple events. We select the latest event that the RTLS sequence related to in this case for the calculation.

7.3 Model Training

Event remaining time prediction is a prediction or regression task. While the input data has similar structure compared with event identification, the goal of the prediction is to give a continuous output value of the time left.

7.3.1 Defining Prediction Target

The target of the prediction is the remaining time of the ongoing event. For model training, this information is extracted from CRF data as we have the event intervals in the CRF data.

For example, suppose we aim to predict the remaining time of event type e, we have a RTLS data with time stamp 9:00:00 am if it is single point data or with window end time stamp 9:00:00 am if it is sequential data. At 9:00:00 am there is an event of type e ongoing, which started at 8:40:00 am and is going to finish at 9:20:00 am, then the prediction target here is 20 minutes or 1200 seconds.

However, the target definition for sequential data can be tricky. The approach of defining target for single point data and sequential data is introduced in the following sections.

Defining Target for Single Point Data

For the same reason as adding event elapsed time, target definition for single point data is relatively straightforward. We take the time stamp of the transient location data item and look up CRF data in the database to find:

$$\{x | T_{start}(x) \leqslant T(l) < T_{end}(x), type(x) = e\}$$

Similarly, As in the filtering process introduced in 7.2.1, we already filtered out the single point data items which do not have relationships with ongoing events and according to Section 6.1 there can be only one event of a specific type going on at the same time point, thus this query will always return one and only one event.

Suppose the event we have found is x, the calculation of the remaining time for the single point data is depicted in equation 7.3

$$remaining\ time_l = T_{end}(x) - T_{end}(l) \tag{7.3}$$

Defining Target for Sequential Data

However, for defining target for sequential data, as introduced Section 7.2.1, there can also be situations other than the example given above. According to the situations listed in Figure 7.1, we list the method of defining target for each of the cases.

According to Figure 7.3, we do not need to take case c and e into consideration because they are already filtered out in the filtering process introduced in 7.2.1. For the other cases, suppose the matching event is x we calculate using equation 7.4

Figure 7.3: Defining Prediction Target for sequential RTLS data

$$remaining\ time_l = \begin{cases} T_{end}(x) - T_{end}(l) & \text{if } T_{end}(x) - T_{end}(l) \geqslant 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.4}$$

In addition, different from defining target for single point data, a sequential data item can possibly relate to multiple events. We select the latest event that the RTLS sequence related to in this case for the calculation.

Target Normalization

Depending on the distribution of the remaining time of a specific type of event, we choose whether to apply normalization or how to apply normalization to the prediction target. The goal is to make the distribution of the normalized target as even as possible.

7.3.2 Algorithm Selection

We select commonly used regression algorithms which can be applied on single point data and sequential data.

The regression models to apply on single point data are listed below:

- Decision tree regressor
- Random forest regressor
- Linear regression
- Neural network regressor. Multilayer perceptron (MLP) is adopted in this study

For predicting event remaining time using sequential RTLS data, long short term memory (LSTM) network is used.

7.4 Experiments and Results

This section describes the objectives, setups and results of the experiments related to the data analytics task of event remaining time prediction introduced in this chapter.

7.4.1 Experiment Objectives

The objectives of event remaining time prediction are similar to that of workflow event identification. The objectives in this experiment are listed below:

- Verify that the concept and methodology of using RTLS data to automatically predict the remaining time of ongoing workflow events are correct and feasible.
- Compare the performance between models trained on single point data and models trained on sequential data to see whether the information about the movement sequences can improve the performance of the remaining time prediction.
- Compare the machine learning approach with a baseline to investigate if there is an improvement using the machine learning approach based on RTLS data and how much the improvement is.

7.4.2 Data Set Specification

The data set used in this experiment is similar to the data set used in the event identification experiment. The data sources are described in Section 6.4.2.

The only differences are:

- In the event remaining time prediction task, we only keep the RTLS data collected during the target events.
- Elapsed time of the ongoing event is added as a new feature. The method of calculating this feature is described in Section 7.2.2
- Event remaining time is set as the prediction or regression target instead of event type. The method of deriving this target is described in Section 7.3.1

7.4.3 Experiment Setup

This section introduces the setup of the experiment for event remaining time prediction. It includes an introduction to the experiment data set, data set splitting, evaluation metrics and the calculation of baseline.

Experiment Data Set

As introduced earlier in this chapter, the data structure of event remaining time prediction is very similar to the data structure of the event identification problem. However, the data set is extracted from the second period and the prediction targets are derived from the CRF data.

Table 7.1: Sizes of data in different formats

Data set	Window length	Size
Single point data	-	12,338
Sequential data	5 minutes	9,812
Sequential data	10 minutes	11,103
Sequential data	15 minutes	12,338
Sequential data	20 minutes	13,563
Sequential data	25 minutes	14,716
Sequential data	30 minutes	15,886

As discussed in the Section 6.4.2, the CRF data has problem of missing events. However, as in the event remaining time prediction task we only do prediction on known ongoing events. This problem will not affect the quality of the target of the data set. Thus we use the CRF data's "first incision" and "close incision point" events to form the target exam event.

The size of the data set is smaller than the data set for event identification because we only keep the portion of RTLS data which is related to ongoing exam events. Table 7.1 describes the data sizes for different formats of data.

Training Test, Validation Splitting

Although the size of the data set is smaller compared with the data set for the event identification experiment, we use the same splitting strategy as stated in Section 6.4.3.

Evaluation Metrics

The metrics for event remaining time prediction is relatively simple compared with the binary classification problem. The main metric we adopted is the Mean Absolute Error (MAE) because it shows the average error in seconds when we predict the remaining time of an event. The second main metric: mean improvement is introduced to compare the performance of our models with the baselines.

Mean Absolute Error . MAE is the average of the absolute difference between the predicted finish time and the actual finish time.

MAE is calculated by

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i|$$

Where y_i is the actual remaining time and $\hat{y}_i$ is the predicted remaining time.

Standard Deviation of Absolute Error The standard deviation of the absolute prediction errors is adopted as a secondary metric to measure the stability of the predictions.

Mean Improvement Compared with Baseline As the baseline MAE can be different for single point data and sequential data, we adopt mean improvement compared with baseline as an additional metric which can compare the performance of all the models involved.

The mean improvement compared with baseline is calculated as:

$$MI_m = MAE_{baseline} - MAE_m$$

Where m is a model for predicting event remaining time.

Experiment Baseline

The baseline for this experiment is the Mean Absolute Error (MAE) when we just use a constant remaining time as the predicted remaining time. This constant remaining time is optimized using the training data.

7.4.4 Evaluation Results

In this section, evaluation results are showed and compared for the event remaining time prediction task. In the experiment, we predicted the remaining time of ongoing exams using 5 algorithms based on single point data and LSTM based on both single point and sequential data. The prediction is only done for the processed RTLS data when there the exam events were ongoing. Thus, the size of the data set is much smaller compared with the event identification data set.

The 6 algorithms in single point data prediction are decision tree regressor, Bayesian ridge regressor, SVM regressor, stochastic gradient descent regressor, multilayer perceptron regressor and LSTM.

Baseline MAE

We have set up grid search to find the optimal constant prediction value and the corresponding MAE.

Firstly we compute the baseline for single point data. The result is shown in Figure 7.4. The result shows that when the constant prediction value is set to 1,145 seconds, we achieve optimal MAE which is 1,268.82 seconds. Thus 1,268.82 is the baseline of the event remaining time prediction.

For sequential data, as the size of the windows change, the size of data changes and there are more data with 0 as remaining time. Thus the baselines of sequential data are different. The baseline computation process is described in Figure 7.5

Performance Comparison Between Different Models

We have evaluated the 6 models listed above. Figure 7.6 and Table 7.2 shows the comparison of the 6 models. Note that the single point data is extracted from the 15 minutes sequential data. So they share the same baseline and are comparable.

From the result, we can observe that the LSTM model based on sequential data performed better than conventional machine learning algorithms in the event remaining time prediction task.

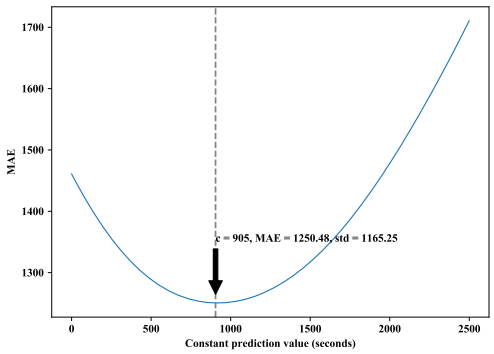


Figure 7.4: MAE when changing the constant prediction value

Table 7.2: Performance comparison between different models

Model name	**Type**	**MAE**	**std**	**MI**
Decision Tree Regressor	single point	1742.88	1669.61	-492.40
MLP Regressor	single point	1333.77	1219.41	-83.29
SVM Regression	single point	1277.94	1197.24	-27.46
Baseline	-	1250.48	1165.25	-
Bayesian Ridge	single point	1227.67	1134.67	22.81
SGD Regressor	single point	-	-	-
LSTM	single point	1245.21	1069.49	5.27
LSTM(15m window)	sequential	**1063.21**	**1126.09**	**187.27**

It improved the prediction accuracy by 187.27 seconds, which is roughly 3 minutes, also the stand deviation is smaller than the baseline. Among the regression models trained with single point data, only Bayesian Ridge Regressor and LSTM performed slightly better than the baseline. SGD Regressor failed to produce meaningful remaining time predictions.

Performance Comparison Between LSTM Models with Different Window Length

We have also carried out experiment on evaluating the performance of LSTM models trained with sequential data of different window lengths. As discussed in the previous section, the baselines for data of different window lengths are different. Thus we put the focus of the comparison on the MAE improvement.

The evaluation result is shown in Figure 7.7 and Table 7.3.

According to the MAE improvement metric, we can observe that when the length of sequential data window increase, the prediction performance becomes higher. When the window length is 30 minutes, we yield mean absolute error of 814.42 seconds (13.6 minutes) which is 301.36 seconds (5.02 minutes) better than the corresponding baseline.

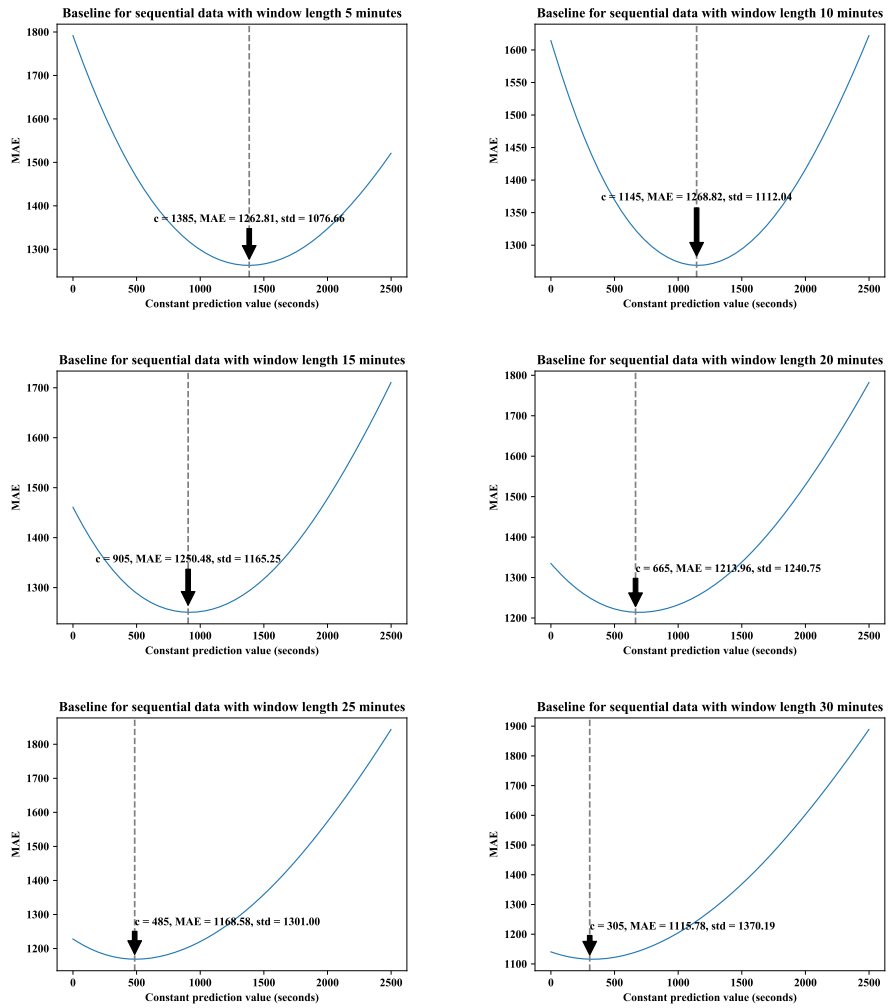


Figure 7.5: Baseline MAE computation for sequential data with different window lengths

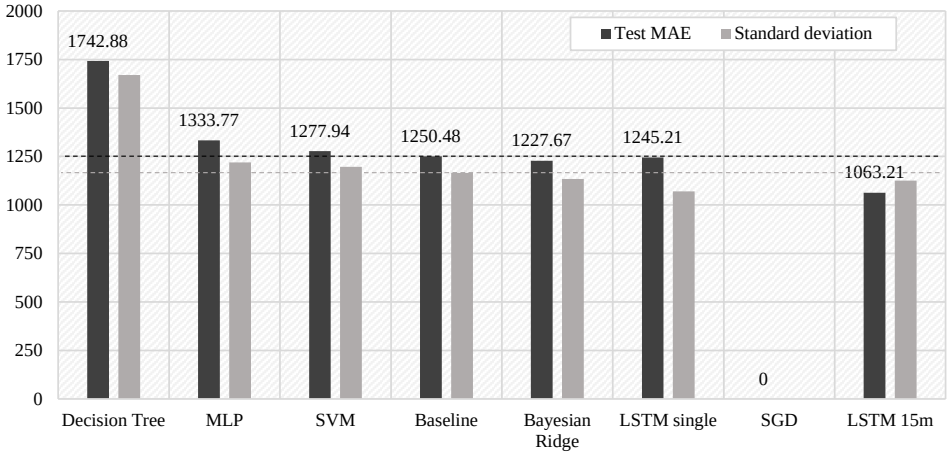


Figure 7.6: MAE and standard deviation comparison on test data for different models

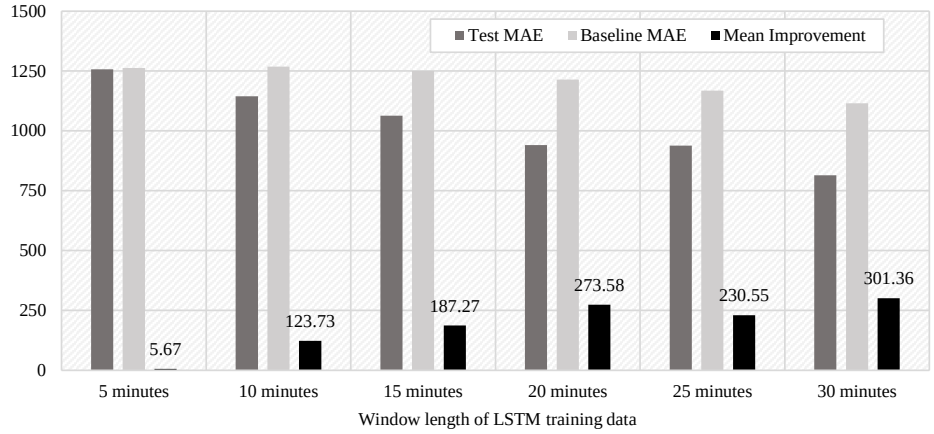


Figure 7.7: MAE, baseline MAE and mean improvement comparison on test data for LSTM trained with sequential data with different window lengths

Table 7.3: Performance comparison between LSTM with different window lengths

Model name	**Window Length**	**MAE**	**std**	**Baseline MAE**	**MI**
LSTM	5 minutes	1257.14	1273.92	1262.81	5.67
LSTM	10 minutes	1145.09	1174.42	1268.82	123.73
LSTM	15 minutes	1063.21	1126.09	1250.48	187.27
LSTM	20 minutes	940.38	1263.87	1213.96	273.58
LSTM	25 minutes	938.03	1400.45	1168.58	230.55
LSTM	30 minutes	814.42	1248.91	1115.78	301.36

7.5 Combining the Power of Event Identification and Remaining Time Prediction

As introduced in the overview of this chapter, the prerequisite for predicting event remaining time given streaming RTLS data is to first know the type of the ongoing event and for how long has the event been ongoing. In the model training and evaluation we construct this data using the existing CRF data. However, our ultimate goal is to build a system which can give hospital real-time predictions on the remaining time of the ongoing events fully automatically. In this case we will not have CRF data to give us the prerequisites because CRF data is manually recorded after events finish.

Fortunately, the works introduced in Chapter 6 could give us the exact prerequisite data we want for making event remaining time prediction. As the event types are automatically identified using machine learning algorithms and we know when is the first time that the current ongoing event is identified, we can calculate for how long has the current event been ongoing. Thus we can adopt the output of the online event identification model to feed event remaining time prediction model. This powerful combination will enable hospitals to not only identify the ongoing workflow events automatically but also predict when is the ongoing event going to finish in real-time.

Chapter 8

Conclusion and Future Work

This chapter concludes the study regarding the three data analytics tasks and the four use cases defined in Chapter 1 and describes the current limitations met in the process of this study. Lastly, future works which could improve this study are discussed.

8.1 Conclusion

Conclusions for the three data analytics tasks are described in this section.

8.1.1 Workflow Playback and Transition Sequence Visualization

With the workflow playback tool, analysts can see an animated tag transitions on a floor plan as well as the progresses of related workflows given a specified time span on an integrated view. The tool has brought intuitive and interactive insights to analysts to understand transition patterns made by staffs in different roles for different types of events. This tool is not only useful for getting an idea of how staffs move around during procedures in an overall picture but also useful for finding interesting transition patterns in particular past procedures.

Different from the workflow playback tool, which only shows the animation of tag transitions in specified short time periods, for example one hour, or an entire shift, the transition sequence visualization tool gives a less intuitive (tag locations are shown on y axis instead on a floor plan) but more general view of how different roles transit and collaborate on in a set of workflow events. Analysts can find interesting common patterns from this visualization which can help them to find potential defects in the current workflows which cause inefficiencies and to find solutions on how to optimize these workflows. For instance, if an analyst observes that nurses are frequently walking back and forth between room A and room B in the middle of percutaneous transluminal angioplasty (PTA) exams, then it might be wise to seek ways to combine the two rooms or shorten the distance between them.

These tools also provided valuable insights for developing the automated workflow event identification and event remaining time prediction models. For instance, we played back a set of exam procedures using the workflow playback tool and observed that physicians are frequently moving back and forth between the control room and the exam room, thus this pattern can be useful for identifying the exam event and we can incorporate these sequential patterns in the features

for workflow identification. Another example is, after the physicians and fellows leave the control room and exit the research area through the corridor, then it is likely that the exam is in wrap-up phase and will end soon. Catching this pattern can be useful for predicting the finish time of ongoing exam events.

Thus, although these two visualization tools are not the main focus of the thesis, they have provided powerful capabilities to clinical analysts and gave us valuable guidance to develop the prediction models.

8.1.2 Automated Workflow Event Identification

The challenge of making classification on RTLS data is that the RTLS data is a type of streaming event data which is not suitable for training machine learning models. To address the challenge, firstly we have proposed an approach of processing and transforming cleaned streaming RTLS data to features to support the machine learning algorithms for making classifications. Secondly, we have proposed an approach to build sequential RTLS data to preserve tag transition information in a time window.

The evaluation results presented in Section 6.4 show that the machine learning models, especially the deep learning model (LSTM) based on sequential RTLS data has achieved high performance on identifying workflow events. After comparing the performances of conventional machine learning and deep learning models based on transient RTLS data and the performances of deep learning models based on sequential RTLS data, we can confirm our assumption that the sequential data contains more predictive power regarding to identify workflow events than transient states.

In addition, we observe that when the window length of the sequential data increase, the performance becomes gradually better. This can be the result of the fact that sequential data with a longer window contains more information in the transition histories. We benefit from this because LSTM has the ability to look back in a large number of steps back in the history.

From the evaluation metrics overview which compares all the machine learning models involved we can conclude that our proposal of identifying workflow events using machine learning models based on RTLS data is feasible. Firstly, AUC and Kappa scores show that the prediction performance is strong. Secondly, the accuracy of 0.84 shows that our model is relatively close to the "gold standard" in identifying a specific workflow event.

The evaluation results imply that our model can be used to identify clinical workflow events based on new RTLS data and achieve relatively reliable result. This model can be potentially deployed in the hospitals with RTLS installed and the staffs tracked to automatically indicate the type of ongoing workflow events in real-time.

8.1.3 Event Remaining Time Prediction

We have proposed a methodology of predicting remaining time of ongoing workflow events. Firstly we proposed an approach of preparing features for event remaining time prediction combining the strategy of producing RTLS features for workflow event identification and features of ongoing events. Secondly we introduced the approach of preparing the prediction target.

The experiments on event remaining time prediction mentioned in Sectioin 6.4 show that machine learning and deep learning algorithms trained with single point data perform poorly in predicting remaining time of ongoing events. However, the deep learning algorithm (LSTM) trained with sequential RTLS data was able to outperform the baseline and provide relatively

accurate predictions on event remaining time.

The evaluation results show that our approach of predicting event remaining time is feasible if we adopt LSTM model with sequential data. The results shown in Section 7.4.4 double confirms that sequential data a longer time window can contribute to better prediction results. The best performing LSTM model improved the MAE by 27% compared with the baseline.

Thus, we conclude that, the LSTM model based on sequential data can be adopted in hospitals with staffs tracked by RTLS and provide better remaining time predictions in real-time. However, there is still space to further improve the prediction precision as the current mean absolute error is still relatively large (13.6 minutes).

8.2 Limitations and Future Work

Firstly, due to the limited data collection period, the prepared data set, especially the data set for remaining time prediction is relatively small considering the need for a large training set for deep learning models. From the result we can observe that there is a positive correlation between the size of the data set and the performance, for instance the MAE improvement in remaining time identification. However, we cannot be sure that an increase in size of the training data will directly lead to a performance increase.

In the future studies, we can collect a longer period of CRF data about more types of workflow events so that we will have a larger data set for training. We could also set up experiments to explore the relationship between data size and prediction performance.

Regarding the data sources, currently we are using only RTLS data because it is the only source from which we can get real-time streaming data. The machine log data, on the other hand, has a few days of delay between the time when logs are produced and the time when we receive them. However, the machine log data contains very comprehensive data about the usage of the iXR. It can add strong predictive power in identifying and predicting exam related events.

The RTLS system we adopted provides zone level precision, thus it is not possible to know the tag's movement inside the zones. In the future, we would like to incorporate RTLS systems which can provide in-room locating capability. This capability can potentially improve the prediction performance because then we will be able to know patterns for instance physician moves from the operation table to the monitor. This information combining the machine log could provide powerful predictive capabilities.

In the future, we could attempt to identify predict remaining time of more types events. After having a complete set of models, we could try to build an online workflow identification and prediction system which can automatically identify new workflows and which steps are the ongoing workflows at and predict the finish time of each remaining step of the ongoing workflows. In addition, the system could be interfaced with the hospital information system to provide scheduling assistance. This feature could improve the efficiency of the hospitals.

Bibliography

[1] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936. xi, 31

[2] Martin LANGab, Thomas Bürkle, Susanne Laumann, and Hans-Ulrich Prokosch. Process mining for clinical workflows: challenges and current limitations. In *EHealth Beyond the Horizon: Get It There: Proceedings of MIE2008 the XXIst International Congress of the European Federation for Medical Informatics*, page 229, 2008. xiii, 6, 7

[3] Karsh B-T. Clinical practice improvement and redesign: how change in workflow can be supported by clinical decision support. *Agency for Healthcare Research and Quality*, June 2009. 1

[4] J. H. Bratt, J. Foreit, P. L. Chen, C. West, B. Janowitz, and T. de Vargas. A comparison of four approaches for measuring clinician time use. *Health Policy Plan*, 14(4):374–381, Dec 1999. 1, 5

[5] K. L. Hunting, E. Haile, L. Nessel, and L. S. Welch. Validity assessment of self-reported construction tasks. *J Occup Environ Hyg*, 7(5):307–314, May 2010. 1, 6

[6] T. L. Jones and C. Schlegel. Can real time location system technology (RTLS) provide useful estimates of time use by nursing personnel? *Res Nurs Health*, 37(1):75–84, Feb 2014. 1, 5

[7] Stewart I. Donaldson and Elisa J. Grant-Vallone. Understanding self-report bias in organizational behavior research. *Journal of Business and Psychology*, 17(2):245–260, 2002. 1, 6

[8] Maged N. Kamel Boulos and Geoff Berry. Real-time locating systems (rtls) in healthcare: a condensed primer. *International Journal of Health Geographics*, 11(1):25, 2012. 2

[9] Alan E Kazdin. Observer effects: Reactivity of direct observation. *New Directions for Methodology of Social & Behavioral Science*, 1982. 5

[10] Rob McCarney, James Warner, Steve Iliffe, Robbert Van Haselen, Mark Griffin, and Peter Fisher. The hawthorne effect: a randomised, controlled trial. *BMC medical research methodology*, 7(1):30, 2007. 5

[11] T. A. Burke, J. R. McKee, H. C. Wilson, R. M. Donahue, A. S. Batenhorst, and D. S. Pathak. A comparison of time-and-motion and self-reporting methods of work measurement. *J Nurs Adm*, 30(3):118–125, Mar 2000. 6

[12] Wil Van der Aalst, Ton Weijters, and Laura Maruster. Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004. 6

[13] Lijie Wen, Jianmin Wang, and Jiaguang Sun. Detecting implicit dependencies between tasks from event logs. *Frontiers of WWW Research and Development-APWeb 2006*, pages 591–603, 2006. 6

[14] AJMM Weijters, Wil MP van Der Aalst, and AK Alves De Medeiros. Process mining with the heuristics miner-algorithm. *Technische Universiteit Eindhoven, Tech. Rep. WP*, 166:1–34, 2006. 6

[15] Gianluigi Greco, Antonella Guzzo, Luigi Pontieri, and Domenico Sacca. Mining expressive process models by clustering workflow traces. *Advances in Knowledge Discovery and Data Mining*, pages 52–62, 2004. 6

[16] Gianluigi Greco, Antonella Guzzo, Luigi Pontieri, and Domenico Sacca. Discovering expressive process models by clustering log traces. *IEEE Transactions on Knowledge and Data Engineering*, 18(8):1010–1027, 2006. 6

[17] Boudewijn F van Dongen and Wil MP Van der Aalst. Multi-phase process mining: Aggregating instance graphs into epcs and petri nets. In *PNCWB 2005 workshop*, pages 35–58, 2005. 6

[18] AA de Medeiros. Genetic process mining (ph. d. thesis). *Eindhoven University of Technology*, 2006. 6

[19] Wil MP van der Aalst, Vladimir Rubin, Boudewijn F van Dongen, Ekkart Kindler, and Christian W Günther. Process mining: A two-step approach using transition systems and regions. *BPM Center Report BPM-06-30, BPMcenter. org*, 66, 2006. 6

[20] David A Freedman. *Statistical models: theory and practice*. cambridge university press, 2009. 8

[21] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. Classification and regression trees. 1993. *Chapman Hall, New York*, 1984. 8

[22] Tin Kam Ho. Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278–282. IEEE, 1995. 9

[23] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001. 9

[24] Stuart Russell, Peter Norvig, and Artificial Intelligence. A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, 25:27, 1995. 9

[25] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985. 9

[26] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. 9

[27] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990. 9

[28] M. I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. *IEEE Press*, pages 112–127, 1990. 9

[29] Ronald J Williams and David Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity. *Backpropagation: Theory, architectures, and applications*, 1:433–486. 9

[30] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011. 15

[31] Thomas G Dietterich. Machine learning for sequential data: A review. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 15–30. Springer, 2002. 15

[32] Centrak. Centrak rtls. `http://www.centrak.com`. 17

[33] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs:(numerische mathematik, 1 (1959), p 269-271). 1959. 23

[34] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011. 28

[35] Alfred Inselberg. Parallel coordinates: visualization, exploration and classification of high-dimensional data. *Handbook of Data Visualization*, pages 643–680, 2008. 31

[36] K. Chang. Parallel coordinates. `https://syntagmatic.github.io/parallel-coordinates`. 32

[37] Receiver operating characteristic. Receiver operating characteristic — Wikipedia, the free encyclopedia, 2017. [Online; accessed 26-June-2017]. 46

[38] Cohen's kappa. Cohen's kappa — Wikipedia, the free encyclopedia, 2017. [Online; accessed 26-June-2017]. 46