Neural Event Prediction for Clinical Event Time-Series

by

Jeong Min Lee

Bachelor of Science, Handong Global University, 2014

Submitted to the Graduate Faculty of

the Dietrich School of Arts and Sciences in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2024

UNIVERSITY OF PITTSBURGH

DIETRICH SCHOOL OF ARTS AND SCIENCES

This dissertation was presented

by

Jeong Min Lee

It was defended on

April 27th 2022

and approved by

Milos Hauskrecht, Committee Chair,

Department of Computer Science, University of Pittsburgh

Adriana Kovashka, Committee Member,

Department of Computer Science, University of Pittsburgh

Erin Walker, Committee Member,

Department of Computer Science, University of Pittsburgh

Shyam Visweswaran, Committee Member,

Department of Biomedical Informatics, University of Pittsburgh

Neural Event Prediction for Clinical Event Time-Series

Jeong Min Lee, PhD

University of Pittsburgh, 2024

Massive clinical event time-series data collected in Electronic Health Records (EHR) offer great potential for improving patient care as they contain in-depth information about patient conditions, relevant diagnoses, and treatment strategies. With event prediction models, we can identify temporal associations among various types of clinical events in EHR, such as symptoms and patient management actions on one side and symptoms and outcomes with or without patient management actions on the other side. Further, we could predict the future occurrence of adverse events and help healthcare practitioners to intervene ahead of time or prepare resources to get ready for their occurrence.

However, building clinical event prediction models has unique challenges posed by inherent characteristics of EHR data: **(1) Different temporal characteristics.** Each event in the multivariate time-series has different temporal behaviors (e.g., repetitively occurring with certain time gaps) and different temporal ranges of dependencies for precursor events. To accurately predict future events from the multiple event time series with different temporal characteristics, we need more flexible and expressive models. **(2) Patient-specific variability.** Based on underlying clinical conditions, each patient's sequence may consist of different sets of clinical events (observations, lab results, medications, procedures). Hence, simple population-based models learned from event sequences for many different patients may not accurately predict patient-specific dynamics of event sequences and their differences.

In this thesis, we propose novel autoregressive event prediction models that can address the aforementioned issues. First, we propose new models that handle different temporal dependencies using multiple temporal mechanisms covering various time scales and temporal behaviors such as recurrence of events and multi-time-scale dependencies. Second, we develop new personalized event prediction models that let us better adjust the prediction for individual patients and their specific conditions. They pursue refinement of population-wide models to subpopulations, patient-specific model adaptation, and a meta-level model switching that can adaptively select the model with the best chance to support the immediate prediction. We evaluate our proposed models on the real-world clinical data derived from EHR of critical care patients. We show that our new models lead to improved prediction performance compared to multiple baselines.

Keywords: Clinical Event Prediction, Sequence Modeling, Event Time-Series Modeling.

Table of Contents

List of Tables

List of Figures

Preface

First and foremost, I would like to express my sincere gratitude to Dr. Milos Hauskrecht, my advisor. This endeavor would not have been possible without his outstanding guidance and support. He not only introduced me to the field of machine learning but also provided great insights for new research ideas in event time-series modelings and clinical data analytics.

I would like to extend my sincere thanks to the defense committee: Dr. Adriana Kovashka, Dr. Erin Walker, and Dr. Shyam Visweswaran. They generously provided knowledge and expertise that helped me to improve the dissertation. I was also fortunate to collaborate with Dr. Joo Heung Yoon who gave me important understandings of critical care medicine.

I also would like to thank fellow members of our research lab, Yanbing Xue, Siqi Liu, Zhipeng Luo, Salim Malakouti, Matt Barren, Zitao Liu, and Ankitkumar Joshi. I wish to extend my special thanks to Keena Walker, our administrative staff, who helped to smooth my life at Pitt.

During my doctoral study, I was privileged to work with amazing colleagues and mentors as an intern and research scientist: Andrey Malevich, Michelle Gong, Komal Kapoor, Yashaswi Alladi, Shuang Yang, and Pei Lin from Meta AI; Even Oldridge, Gabriel de Souza Pereira Moreira, and Sara Rabhi from NVIDIA. I am grateful to have many good friends in Pittsburgh, Seattle, Bay Area, and South Korea who made my life more enjoyable.

Finally, but most importantly, I would like to thank my parents, Deuk-Man Lee and Yun-Ja Jung, for all the prayers, encouragement, and love they gave to me during the entire course of pursuing my Ph.D. All glory to God!

> Surely goodness and mercy shall follow me all the days of my life; And I will dwell in the house of the Lord forever. Psalms 23:6

1.0 Introduction

1.1 Motivation

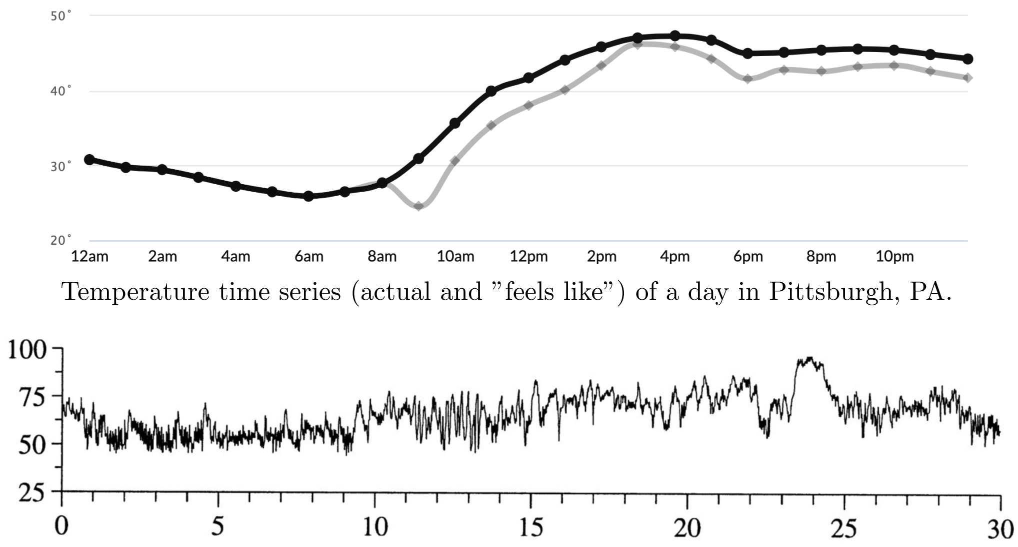
Time series are everywhere around us. They are formed by recordings of observations of either a natural or a man-made system in time, reflecting its temporal behavior and changes. In general, time series may capture the behavior of various physical systems (movement of celestial objects, solar activity, seismic activity, weather), human body (heart rate, blood pressure, and oxygen saturation), human activity (web page viewing logs), or even society (traffic behaviors, crime rates, or election results).

1.1.1 Types of Time Series

Time series can be formed by observations that are made at regular time intervals (or equivalently with a fixed frequency). Examples of such time series are hourly observations of temperature at the weather station in Figure 1a or recording of electrocardiography (ECG) signal in medicine Figure 1b. The measurements in both of these time series are real-valued values. This type of time series is known as *regular* time series, which is typically generated from a device with automated observations at predefined frequency.

However, not all time series are generated at regular time intervals. An example of such a time series are recordings of seismic activities in Figure 2a. An entry is a discrete event recording of the occurrence of an earthquake in time and its magnitude. Another example of such a time series are measurements of Prothrombin Time (PT) lab test over 50 days of a septic patient in Figure 2b. Each PT observation is made only when the measurement event occurs, such as when a physician orders a lab test to assess the underlying physiological conditions of the patient. This type of time series is referred to as *event* time series, and it records occurrences of discrete events with or without a value. Unlike the regular time series the time gaps between the two consecutive data points in event time series can be irregular.

All previous time series examples were *univariate* time series, that is, they consisted of



Temperature time series (actual and ”feels like”) of a day in Pittsburgh, PA.

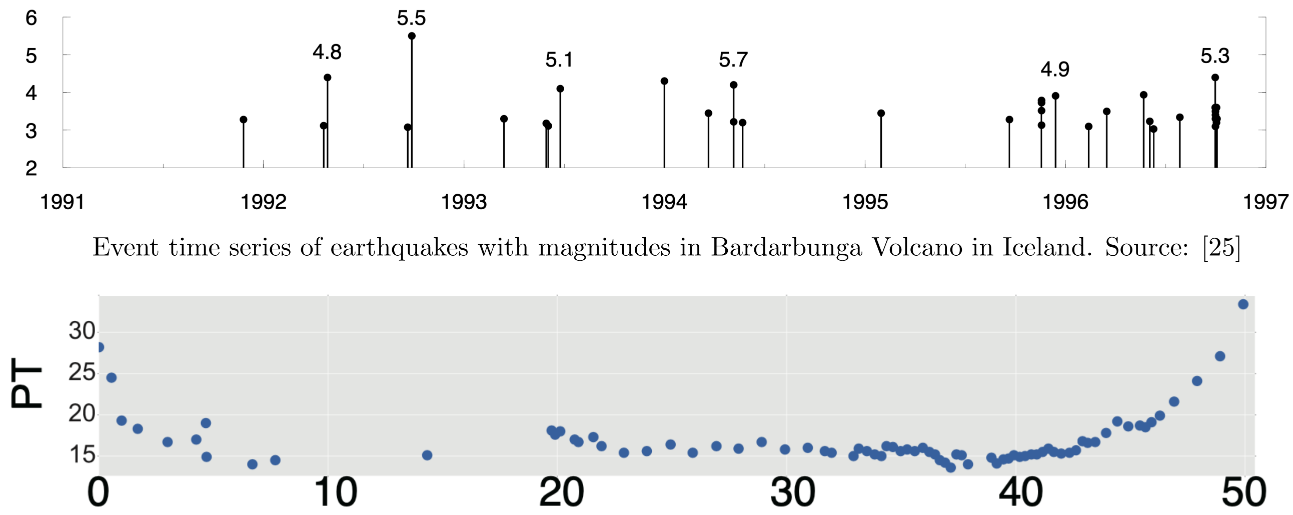
Heart rate time series from a bedside monitoring device (30 minutes). Source: [59]

Figure 1: Examples of regular time series which are generated from devices with automated observations at predefined frequency.

a sequence of measurements of one variable (one data type). However, many time series in real world consist of multiple measurements. Such time series are referred to as *multivariate* time series. An example of such a time series are multiple measurements/signals recording the different aspects of patient's condition in time in Figure 3. Each entry indicates an occurrence of a specific type of clinical events such as administration of medication, lab test order, or medical procedure. As shown in Figure 3, when multiple individual event time series are combined, it may look like a univariate event time series where each entry represents an occurrence of a different type of event.

1.1.2 Tasks Defined upon Time Series

Recent advances in data acquisition and processing technologies enabled collections of massive time series datasets in various domains. In manufacturing, for example, it is common for factories to have multiple sensors monitoring the manufacturing process and they generate hundreds of thousands of sensor readings per day. In finance, time series data is generated



Event time series of earthquakes with magnitudes in Bardarbunga Volcano in Iceland. Source: [25]

Prothrombin time (PT) test time series of 50 days from a septic patient. Source: [33]

Figure 2: Examples of event time series which record occurrences of discrete events. Time gaps between two consecutive data points can be irregular.

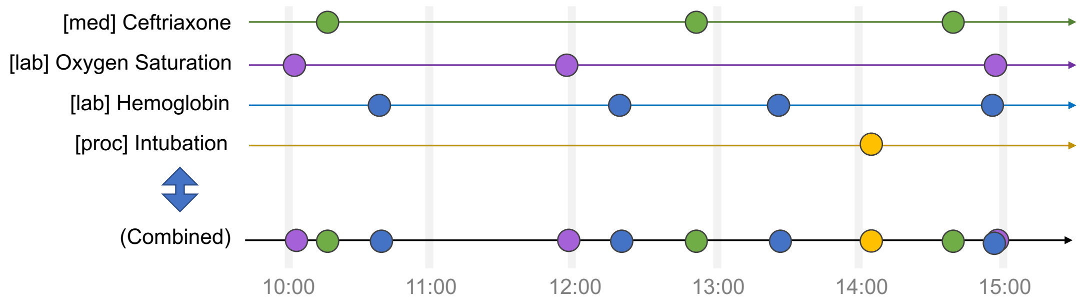


Figure 3: Example of multivariate event time series from Electronic Health Records (EHR)

from financial transactions and changes of stock market prices from all over the world. In healthcare, patient electronic health records, wearable devices, and continuous monitoring devices generate massive clinical time series data. In transportation, GPS tracking systems can generate time series data describing the location of vehicles and/or people in real-time. Such massive collections of time series data provide new opportunities to use the data to solve multiple important tasks in respective domains. These may cover monitoring of the dynamic system, detection of special behaviors, e.g. malfunctions of the system, or prediction of the future state of system. In the following, we outline four basic tasks or problems one can define in context of time series:

- **Time series classification** aims to assign a label (a category) to a time series that helps us to distinguish it from other similar time series. The label may indicate a different pattern or a condition associated with the time series, such as a season, a weather pattern (storm, rain, cloudy) for weather-related observations [201], or an interpretation of a heart condition for an EKG time series signal [121].
- **Time series clustering** aims to group individual time series based on their similarities among a set of multiple time series. Trends of observed values in time series, event occurrence patterns, or the types of observed events can be used to measure the similarity. For example, with biomarkers time series data from Parkinson's Disease patients, we can identify groups of patients who have similar disease progression patterns [226]. These patient time series clusters can be advantageous to design treatment plans bespoke to similar groups of such a disease with heterogeneous subtypes.
- **Time series forecasting** aims to predict the future values of a time series based on past observed values and their historical trends. Modeling both the overall trajectories of a time series and the future value's dependencies on recent changes in the time series is vital to making an accurate forecasting. This has been an important research topic for extensive application problems in many domains, such as stock price prediction in financial industry [4, 106], temperature prediction in meteorology [42], future lab measurement prediction in preventive health care [129].
- **Event prediction** aims to predict the occurrence of next event and other information associated with the event, such as the event type and/or timing of the next event. Al-

though this problem is closely related to time series forecasting, the difference is that event prediction focuses on predicting discrete event occurrences and modeling dependencies between event occurrences. Examples of event prediction problems are user activity event modeling in online recommendation systems [44, 172], crime event prediction in law enforcement operations, or predicting adverse clinical events like a septic shock for early warning systems in hospitals [58, 157].

1.1.3 Machine Learning Solutions

In recent decades, machine learning has become a powerful tool to solve various problems due to its ability to learn from experiences recorded in the data. In terms of time series, it is able to learn complex temporal associations and dependencies in the time series data needed to support the above tasks. For example, we can solve the time series classification problem for EKG time series by training a machine learning model that learns to map each EKG time series to the corresponding heart condition (class label). Through the learning process, the model obtains a capability to extract key characteristics of time series, such as, peaks, gaps between peaks, and trends of the sequence of observed electric signal measurements, that can be effective in distinguishing different heart conditions. Afterward, when a new patient's EKG time series is given, the model can adequately assign the heart condition (label) to the patient.

Similarly, the time series forecasting problem can be solved by training a machine learning model that learns historical trends and patterns of time series data by associating a series of preceding observations with the newly observed value. For instance, we can forecast the next day's temperature by training a model that learns the association between each day's temperature and the last several preceding days' temperatures based on historical records of a city's daily temperature for the last several years.

In terms of event prediction problem we want to learn the dependencies between an event occurrence and a series of other events that occurred before the event. For example, with event time series data of user web browsing history from an online e-commerce website, we can train a machine learning model that predicts a product that a user is likely to click in the near future based on the user's recent browsing history. The dependencies between different items can be learned through the model training process: on the e-commerce website, with many user sequences of visiting a camera product followed by visiting a camera accessory page, the model can learn the association between these two products. Then, for those who recently visited the camera page, the model can adequately recommend camera accessories that can be relevant.

Finally, for the time series clustering problem, we can train a machine learning model to identify subgroups of time series instances such that distances between the time series instances within the same subgroup are minimized, and distances to time series in the other subgroups are maximized. For this, defining and measuring distances between time series is an important issue and many approaches have been studied. For example, with many patient sequences of medication administration events, we can discover patient subgroups such that patients with similar medication administrations are grouped by training a clustering model. In this case, the number of the common type of medications can be used as a similarity measure between different patient event time series instances.

Besides the advantages mentioned above, machine learning also comes with another important benefit: the models can be gradually refined when more data examples become available. In general machine learning models tend to improve their performance when considering and using more data.

1.1.4 Time Series in Healthcare

One of the areas that generate an abundance of interesting yet complex time series data is healthcare. Our ability to analyze such data, and develop machine learning models and solutions based on these data is extremely important since it may directly impact patient management and consequently patients' physical and mental well-being. A wide range of time series data are generated in health-care settings. These include data generated by various clinical and health-assisting devices such as bedside monitoring systems [99, 142, 193] or smart healthcare solutions based on smart watch [30, 133], smart phone [107, 208], internet of things [60, 68], and social media [115, 148, 149]. In this thesis, we focus our interest on **event time series** data from **Electronic Health Records (EHR)**, a comprehensive database of patient-related measurements, observations, and treatments that reflect patient conditions, their management, and their dynamics. In terms of a problem, the main focus of this thesis is on **event prediction**. By applying event prediction to EHR-derived event time series data, we can build powerful machine learning models that can perform a variety of practical tasks such as predicting patient outcomes (e.g., mortality, readmission), predicting patient management actions (e.g., medication orders, lab tests), or predicting adverse clinical events (e.g., hypotension, septic shock).

1.2 Electronic Health Record (EHR) Data and Challenges

Data in EHR are invaluable assets to improve patient care as they contain in-depth information about the patient's conditions, relevant diagnosis, treatment strategies, and prognosis. Each time patients are engaged in their medical care, detailed information about the care is collected through various sensors and devices in the hospital. This information is aggregated and stored in EHR. Hence, EHR contains various types of patient data such as records of symptoms, order and administration records for medications, lab test orders and results, types of procedures performed, records of physiological signals from bedside monitoring devices, administrative codes, clinical notes written by physicians, and other clinical information.

From the perspective of event time series data, each data entry in EHR can be considered as an event about a patient. For example, when a new clinical event occurs during patient care (e.g., a doctor orders medication for a patient), the new event is recorded in EHR with timing information as well as attributive information such as the type of event (e.g., medication administration), the item involved in the event (e.g., name of the medication), and the value associated with the event (e.g., the dosage of the medication administered). For instance, it takes a form of tuple in database like the following: `(patient id:4980, timestamp:2022-04-01 13:05, event type:'medication order', item:'insulin', volume:20ml)`. Collectively, when we look at these numerous clinical events associated with

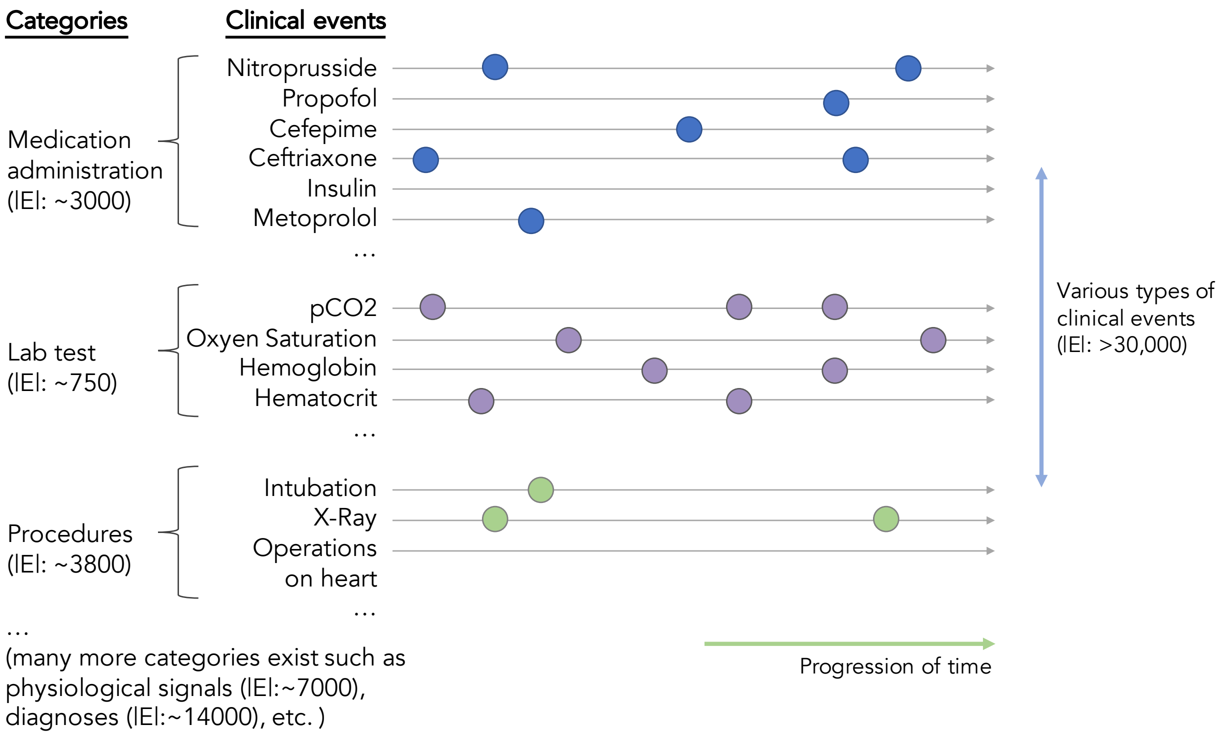


Figure 4: Illustration of a patient's clinical care history in electronic health records (EHR). The history is represented as multivariate event time series. A circle on time-axis corresponds to an occurrence of a clinical event. The numbers of clinical event in each category are counted from MIMIC-3 Database.

a patient on the axis of time, we indeed observe a complex *multivariate* event time series where each type of event forms a univariate event time series as shown in Figure 4.

Building machine learning models for EHR data has the potential to improve and advance patient care beyond traditional methods. For example, we may be able to identify and explore temporal relationships among various types of clinical events, such as symptoms and patient management on one side and symptoms and outcomes with or without management interventions on the other. Further, we could predict the future occurrence of adverse events and help healthcare practitioners to intervene ahead of time or prepare resources to get ready for their occurrence. All of this, in turn, can improve the quality of patient care [26, 31, 87, 222].

However, building machine learning models from EHR data poses several challenges due to its unique characteristics. In the following, we briefly discuss the challenges which will be addressed throughout this thesis:

- **High dimensionality.** Since EHR aggregates almost every type of data about patient care and patient management in hospitals, there are tens of thousands of different types of clinical events that could occur for a patient at any time during the hospitalization. For a machine learning model, the model needs to learn and maintain the knowledge about each event type (as a certain internal representation) for all different event types. Furthermore, in order to properly predict future events, the model needs to learn complex dependencies between a future event and a series of other events that occurred before it. However, almost countless combinations of these past-future event pairs exist, and it can be a great computational challenge to enumerate and learn these combinations. For example, if we have N different types of events that could (co-)occur at any point of time and we want to model past occurrences of events in the last K time steps to predict the next event, there exist $(2^N)^{K+1}$ different past-future event combinations. Even for a simple case where we have (N=) 1000 types of different events and we model past $K = 2$ steps, these combinations are beyond the number of atoms in the observable universe.
- **Missing and irregular observations.** Ironically, although we have tens of thousands of different event types in EHR, many events are not observed in every patient data. This is because many types of clinical events are primarily tied to specific diseases or

complications experienced by a patient. For example, clinical events about insulin administration are most likely observed only in patients with diabetes. In the case of the MIMIC-3 database[1] [89], more than $30,000$ different types of clinical events exist, but the average number of occurrence[2] of each event type across different event categories is usually very small such as medication administration (10.1), lab tests (7.3), and procedures (1.5). This missing observations are challenging because it is hard to train a robust model that can perform well for unseen patients in a test (or validation) set without having enough training instances for all different event types. In addition to missing observations, another challenge of EHR data is that time series observations for many data events are not collected regularly with a specific frequency, instead, the gap between the two consecutive observations or events may vary.

- **Patient variability.** Finally, another important challenge of developing machine learning models for EHR-derived data is the heterogeneity of patient sequences across patient population. Typically, clinical event sequences in EHR are generated from a pool of diverse patients where each patient has different types of clinical complications, medication regimes, or observed sequence dynamics. While the average behavior of clinical event sequences can be captured well by a single machine learning model, the machine learning models may fail to represent the detailed dynamics of heterogeneous clinical event sequences for individual patients.

In order to fully utilize EHR data, it is essential to address and resolve the issues. Hence, throughout the thesis, we focus on developing efficient and scalable methodologies that can address these challenges. More details of these challenges and our approaches to them will be discussed in Section 1.4.

[1] MIMIC-3 is one of the widely-used publicly accessible EHR for research usage. It contains de-identified 53,423 distinct hospital admissions records from Beth Israel Deaconess Medical Center in Boston, Massachusetts, collected between 2001 and 2012

[2] We computed the average counts from the following tables in MIMIC-3: `inputevents-mv`, `labevents`, `procedureevents-mv`

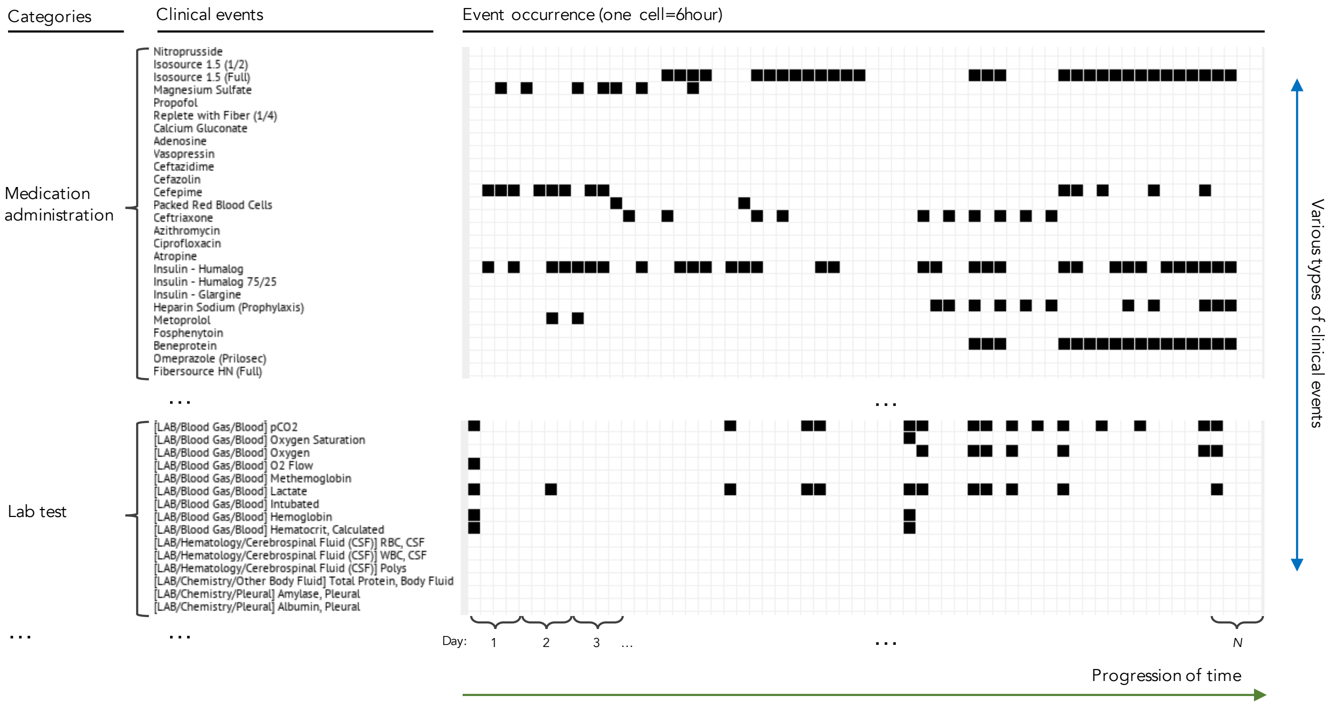

Categories
Clinical events
Event occurrence (one cell=6hour)
Medication administration
Nitroprusside
Isosource 1.5 (1/2)
Isosource 1.5 (Full)
Magnesium Sulfate
Propofol
Replete with Fiber (1/4)
Calcium Gluconate
Adenosine
Vasopressin
Ceftazidime
Cefazolin
Cefepime
Packed Red Blood Cells
Ceftriaxone
Azithromycin
Ciprofloxacin
Atropine
Insulin - Humalog
Insulin - Humalog 75/25
Insulin - Glargine
Heparin Sodium (Prophylaxis)
Metoprolol
Fosphenytoin
Beneprotein
Omeprazole (Prilosec)
Fibersource HN (Full)
...
Lab test
[LAB/Blood Gas/Blood] pCO2
[LAB/Blood Gas/Blood] Oxygen Saturation
[LAB/Blood Gas/Blood] Oxygen
[LAB/Blood Gas/Blood] O2 Flow
[LAB/Blood Gas/Blood] Methemoglobin
[LAB/Blood Gas/Blood] Lactate
[LAB/Blood Gas/Blood] Intubated
[LAB/Blood Gas/Blood] Hemoglobin
[LAB/Blood Gas/Blood] Hematocrit, Calculated
[LAB/Hematology/Cerebrospinal Fluid (CSF)] RBC, CSF
[LAB/Hematology/Cerebrospinal Fluid (CSF)] WBC, CSF
[LAB/Hematology/Cerebrospinal Fluid (CSF)] Polys
[LAB/Chemistry/Other Body Fluid] Total Protein, Body Fluid
[LAB/Chemistry/Pleural] Amylase, Pleural
[LAB/Chemistry/Pleural] Albumin, Pleural
...
...
Day:
1
2
3
...
...
N
Progression of time
Various types of clinical events

Figure 5: A part of a patient's record in real-world EHR (MIMIC-3 database) represented as a sparse matrix. Rows correspond to different clinical events and columns correspond to time. Each cell (bin) indicates occurrence or non-occurrence of an event during a time-window (e.g., 6 hours).

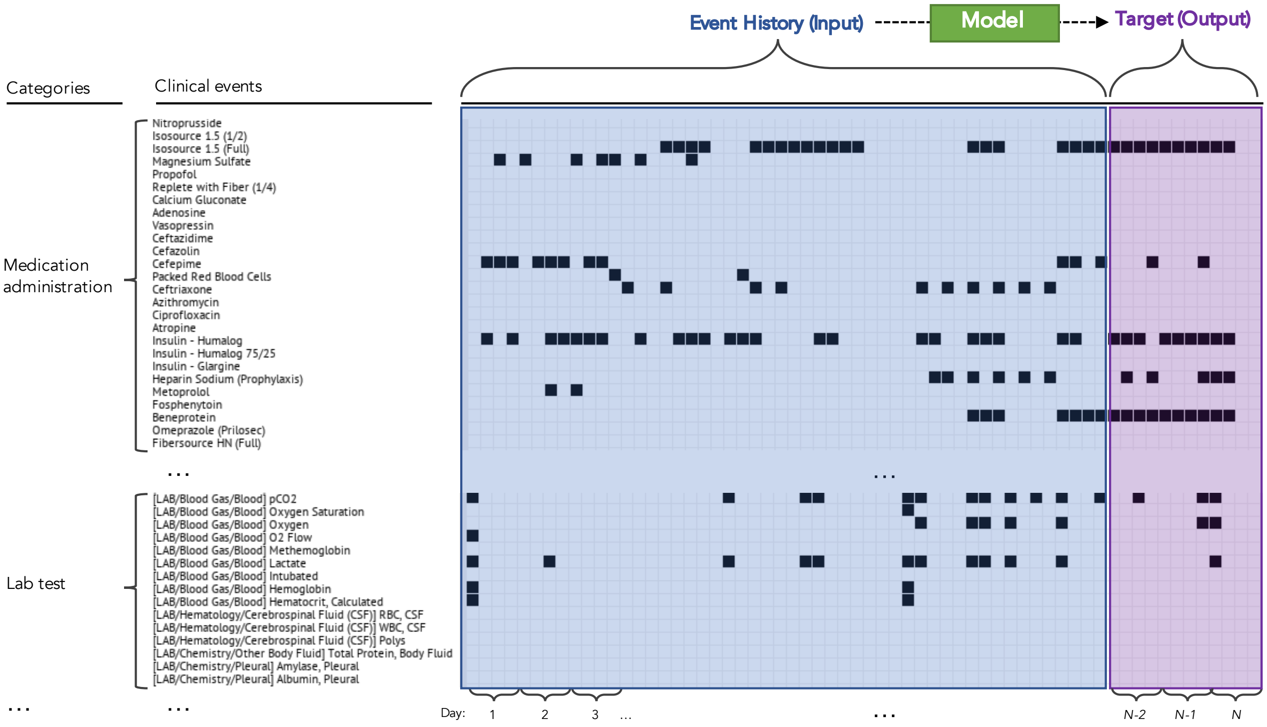

Event History (Input)
Model
Target (Output)
Categories
Clinical events
Medication administration
Nitroprusside
Isosource 1.5 (1/2)
Isosource 1.5 (Full)
Magnesium Sulfate
Propofol
Replete with Fiber (1/4)
Calcium Gluconate
Adenosine
Vasopressin
Ceftazidime
Cefazolin
Cefepime
Packed Red Blood Cells
Ceftriaxone
Azithromycin
Ciprofloxacin
Atropine
Insulin - Humalog
Insulin - Humalog 75/25
Insulin - Glargine
Heparin Sodium (Prophylaxis)
Metoprolol
Fosphenytoin
Beneprotein
Omeprazole (Prilosec)
Fibersource HN (Full)
...
Lab test
[LAB/Blood Gas/Blood] pCO2
[LAB/Blood Gas/Blood] Oxygen Saturation
[LAB/Blood Gas/Blood] Oxygen
[LAB/Blood Gas/Blood] O2 Flow
[LAB/Blood Gas/Blood] Methemoglobin
[LAB/Blood Gas/Blood] Lactate
[LAB/Blood Gas/Blood] Intubated
[LAB/Blood Gas/Blood] Hemoglobin
[LAB/Blood Gas/Blood] Hematocrit, Calculated
[LAB/Hematology/Cerebrospinal Fluid (CSF)] RBC, CSF
[LAB/Hematology/Cerebrospinal Fluid (CSF)] WBC, CSF
[LAB/Hematology/Cerebrospinal Fluid (CSF)] Polys
[LAB/Chemistry/Other Body Fluid] Total Protein, Body Fluid
[LAB/Chemistry/Pleural] Amylase, Pleural
[LAB/Chemistry/Pleural] Albumin, Pleural
...
...
...
Day:
1
2
3
...
...
N-2
N-1
N

Figure 6: Prediction task defined over the multivariate clinical event time series introduced in Figure 5. Given full event history (blue box), the goal is to predict occurrences of each events in future window (purple box).

1.3 Clinical Event Time Series Prediction

In this section, we introduce the problem of clinical event time series prediction and the patient state representation learning, which is an important component of the prediction problem.

Briefly, events in event time series occur in continuous time and in statistics, they are modeled as temporal point processes, that is, point processes defined on time dimension [82, 103, 178]. The basic temporal point process defines the occurrence of just one type of event. Marked point processes associate values with each event occurrence [84, 102]. If values associated with events are categorical, they represent multivariate event processes. That is, each event category defines its own basic point process [122]. With EHR data, we can obtain multivariate event time series for each patient by representing each clinical event occurrence, such as administration of a medication, as an event in the time series data.

Based on the EHR-derived multivariate event time series data, we can define the event prediction task as follows: given a full history of events in a sequence $y_{[1:t]}$ (from the beginning until current time t), the task is to predict the occurrence of the next (future) event y_{t+1}. For continuous-time prediction, this is typically done by defining and modeling an intensity function of the point process. Hawkes process models [104, 180] or its variants [122, 123, 152] can be used for this case.

However, instead of defining and learning the intensity function for continuous-time prediction, one may also convert the time domain of the event time series from continuous time to discrete time by discretizing the time series, and restricting predictions to a finite time interval. As shown in Figure 8, we can discretize the time series by having a (non-overlapping) moving window over the event time series, and representing the same type of events that occurred within a time window (e.g., 6 hours) as a binary indicator value. With this discretization, the multivariate event time series can be represented as a large, sparse binary matrix like what is shown in Figure 5. Different rows in the matrix correspond to different event types, and columns correspond to segmented time steps during the patient's hospitalization. Likewise, the prediction task can be defined over the sequence of column-vectors of the matrix as shown in Figure 6. A detailed definition of the multivariate event time series and a formal denotation of the event prediction task will be presented in Chapter 2.

1.3.1 Patient State Representation for Clinical Event Prediction

While developing event prediction models for EHR-derived data, one of the most important challenges is to summarize past patients' history in EHR such that the summary is pertinent for the next event prediction. In this context, the information that concisely summarizes a patient's history important for prediction is often referred to as **patient state**. Developing efficient and effective methods that can generate patient states is essential to developing powerful clinical event prediction models. In what follows, we briefly introduce existing approaches to define the patient state and corresponding prediction models for EHR-derived multivariate event time series.

Recent observations. One straightforward way to define patient state is to use the most recent observations. For example, given a patient's longitudinal event history from admission up until now, we can only use recent observations such as what is observed during the last 6 hours and ignore other observations before it. This simple method can be effective since recent observations are more likely similar to what will be observed in the near future, compared to observations made in farther past. This method also is efficient since it only uses a handful of recently observed values, and this could eliminate the amount of computation otherwise needed to process the entire patient history. Another important advantage of this approach is that we can use most off-the-shelf classification algorithms such as support vector machines (SVM), Naive Bayes classifiers, decision trees, or neural networks for future event prediction. With the most recent observations for each event type, we can create a vector that size of the all event types and feed the fixed-sized vector to the off-the-shelf classification algorithms. However, this approach has drawbacks: (1) This can restrict the model from learning long-term trends or event dependencies over longer pasts. Furthermore, the patient's condition can change rapidly over time and a recent observation may no longer be representative of the current condition. In this case, using recent observations may not be able to accurately capture the patient's underlying physiological condition. (2) It misses information about those types of events that have their last occurrence in a long past, since this approach only considers what is observed in the recent past.

Last Value Carry Forward. This method addresses the second drawback of the previous approach by copying a value from the latest occurrence for each event type and using it as the current patient state. As a simple yet effective approach, this has been a popular method for handling missing data in clinical and medical studies that involves longitudinal data [65, 131, 154]. Same as the previous approach, this method also can be used with any classification methods for next event prediction. However, the complex trends or event dependencies over longer pasts still cannot be properly modeled with this approach since this approach still uses the latest observations for each event type as the input to the event prediction model.

Temporal Templates. This method solves the aforementioned issues through predefined temporal templates (featurization) of individual time series and their combinations [71, 205]. Briefly, the temporal template approach transforms complex multivariate clinical time series with either discrete or real values in long pasts into fixed-sized vector representations. The gist of the method is to define a set of feature functions (also called feature templates) that map time series defined over clinical variables to fixed-size vectors and their combinations [71]. Examples of the feature functions are event-type-specific summary statistics such as minimum, maximum, or average of the observations over certain time windows (e.g., last 6, 12, and 36 hours) for real-valued time series, or counts of event occurrences for discrete event time series. Since it can provide a more comprehensive summary of clinical time series data, many early works on predicting clinical events from EHR data relied on the templates approach. The fixed-size feature vectors from the templates are fed to any classification algorithms to make a prediction for the next event. This has been successfully used for different EHR prediction [162, 204] and outlier detection [69, 70] problems. However, the main disadvantage of the approach is that temporal templates and info they represent should be defined a priori, and the number of possible features generated with these methods can be very large. One solution to alleviate the need to define the templates a priori is to use predictive patterns extracted directly from data using frequent data mining methodologies [10, 14, 17].

Probabilistic Latent State-Space Models. More recent works have focused on defining the patient states and predictions using various probabilistic latent state-space models such as hidden Markov models, linear dynamical systems [126, 128], Gaussian processes [101, 185], or their combinations [125, 130]. This approach allows more flexibility by modeling complex dynamics of the clinical time series through a (shared) *latent* state-space, which is defined by an autoregressive function of a previous latent state and a recent observation. The benefit is that correlated observations can be represented more compactly in the latent space. A limitation of probabilistic models is that the behavior and expressiveness of the latent state-space are determined by a specific (predefined) probabilistic distribution such as Gaussian distribution, Bernoulli distribution, or Weibull distribution, which may not exactly fit the observed data.

Modern Neural-based Models. Most recently, advances in modern latent embedding and deep learning models led to new low-dimensional latent state representations with good predictive performances on a variety of tasks. Examples of the relevant works include modeling of a patient state using real-valued vector-based representation methods such as Skipgram and CBOW [36, 50, 53, 140, 138, 139], hidden state-space models based on recurrent neural networks (RNN) [8, 35, 38, 51, 86, 108, 109, 110, 111, 112, 113, 169, 224], or non-recurrent models such as attention mechanism, convolutional neural networks (CNN), and Transformers [37, 46, 117, 159, 175, 177, 179, 195, 227]. These modern approaches typically do not assume a specific probabilistic distribution form for generating the hidden state space. Instead, they use a data-driven approach to learn the mapping of the input to the hidden state and ultimately to the output using a series of linear transformations (matrix multiplications) and non-linear activation functions (e.g., sigmoid or tangent hyperbolic). Hence, it is typically more flexible (no specific distribution form is assumed) and more capable of learning non-linearities lie in the complex EHR-derived time series data compared to the probabilistic latent-space approaches. In this thesis, we build upon and explore models based on modern neural-based temporal methods.

Hence, the primary focus of this thesis is to develop methods that can learn good patient state representations and corresponding event prediction models for complex EHR-derived multivariate event time series data.

1.3.2 Clinical Relevance of the Event Prediction Models

In this thesis, we develop event prediction models that can predict occurrences of a broad range of events in EHR. These models can be used for different clinical purposes. If events predicted are equal to adverse events, our ability to predict boils down to adverse event predictions. Examples of such problems are predictions of sepsis [73, 157] or acute kidney injury (AKI) [95]. However, we would like to note that some adverse events may not be directly logged in the EHR. In that case, surrogate events and conditions can be used to define these events and enrich the EHR data with augmented event sets. For example, one may define the sepsis event by the time when the standardized Sepsis-3 definition is satisfied [192]. Similarly, AKI prediction targets can be incorporated into EHR using AKI definitions based on the serum creatinine levels and urine output [21, 94, 151].

Our event prediction models can also be used for outlier detection, and medical error detection as defined in the works of Hauskrecht et al. [69, 70, 71, 72, 124]. Briefly, by defining high-quality models for predicting the events like lab orders or medication administration, one can use them to infer unexpected omission or commission of medications or labs. Finally, our ability to predict the occurrence of future events for multiple patients at the same time can be used to predict various future resource demands, which in turn can be used to optimize the workflows or predict various capacity limits [92, 141, 156, 223].

1.4 Research Goals and Hypotheses

As briefly mentioned earlier, EHR-derived multivariate event time series data pose unique challenges for the development of corresponding event prediction models. The goal of this thesis is to address and develop solutions for some of these challenges. In particular, we focus our investigations on two major research questions (RQ):

- **RQ1.** How to learn effective patient state representation and transitions while modeling unique characteristics of EHR-derived event time series data?
- **RQ2.** How to learn a personalized and adaptive patient dynamic representation that

can address the variability of the heterogeneous individual patient event sequences?

In the following, we discuss these research questions in more detail.

1.4.1 Research Goal 1: Learning Patient State Representation and Transitions

Our first goal is to develop effective methods for patient state representation and transitions for EHR-derived multivariate event time series. First, we focus on the inherent characteristics of EHR-derived event time series and develop event prediction models addressing them. Following are the specific challenges we want to deal with in this thesis.

1.4.1.1 High Dimensionality

Multivariate event time series for hospitalized patients consist of several thousands of different types of clinical events. For example, they correspond to the administration of many different medications, lab orders, lab results, various physiological observations, procedures, etc. For instance, as mentioned in Section 1.1, more than 30,000 different types of events exist in the MIMIC-3 Database. When representing multivariate event time series into a matrix, such as Figure 5, it becomes a large, sparse matrix and the complexity from it may not fit standard statistical time series models [119] with either observed or hidden state transition models.

To address these issues, some works attempted to predict a singly occurring target event (i.e., one type of target event), instead of predicting full-multivariate events as the target [38, 50, 51, 86, 159, 169]. In contrast, **we aim to predict high-dimensional targets from the sequence of high-dimensional events**. It is more challenging as the models need to learn more complex associations between context and target over multiple steps of time.

1.4.1.2 Time-Representation and Temporal Granularity

The original EHR-based multivariate event time series consists of events recorded on continuous-time. To efficiently process the time series, the original continuous-time repre-

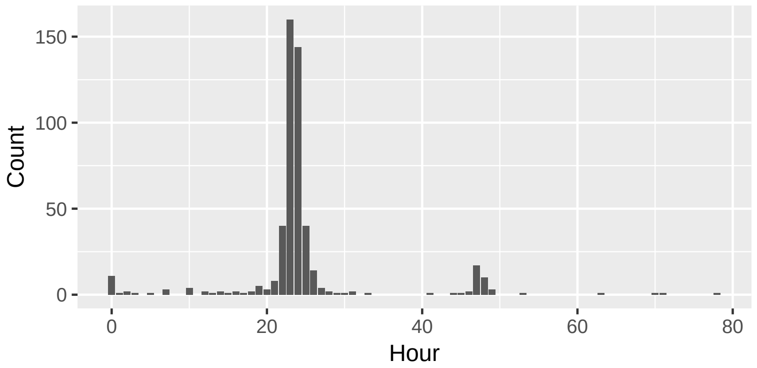


Figure 7: Histogram of time differences for two consecutive events of administration of antibiotics medication, Fluconazole. It illustrates how events in EHR occur with periodicity.

sentation is typically processed to discrete-time based representation using window-based segmentation [69, 70, 108, 175] which maps multiple events that happen during a specific time-window in a fixed-sized multi-hot vector. During the segmentation process, the derived event time series can be generated at a certain temporal granularity which corresponds to the size of the window. Finer temporal granularity results in the detailed representation of patient states in high resolution. But at the same time, it incurs a challenge of longer and sparser sequences which make modeling dependencies over time harder. Also, computationally it is more expensive.

To avoid these issues, some of the prior works on modeling EHR-based event time series used coarser temporal granularity such as admission or visit levels [35, 50, 51]. In this thesis, **we consider and build models with event time series based on finer temporal granularity** such that each window (a time-step in a sequence) in the derived event time series summarizes 6, 12, or 24 hours of the original continuous-time event time series.

1.4.1.3 Heterogeneous Temporal Characteristics

The EHR-based multivariate event time series consists of individual event time series that have heterogeneous temporal characteristics. For example, some types of events occur repetitively with certain time gaps (e.g., medications administered at regularly scheduled intervals, as shown in Figure 7). Also, each event has different temporal ranges of dependencies for precursor events. Some events are strongly dependent on very recent occurrences. For example, observation of administration of phenylephrine (a medication that increases blood pressure) could highly relate to an observation of hypotension (low blood pressure state) in close prior time. In other instances, events may depend on a preceding event that occurred a long time before. For example, the incidence of acute kidney injury (AKI) in the distant past can impact the necessity of kidney dialysis. To accurately predict future events from the multiple event time series with different temporal characteristics, we need more flexible and expressive models. In this thesis, **we focus on developing methods that can model different temporal characteristics with its modularized architecture**. Specifically, we develop models that consist of a set of modules where each focuses on a specific temporal attribute. With this approach, we can build an expressive and flexible ensemble model for multivariate time series prediction.

1.4.2 Hypotheses for Research Goal 1

To alleviate the aforementioned challenges in the first goal, we propose new autoregressive neural temporal models that can handle complex multivariate event time series with more expressiveness by equipping different information channels for various temporal characteristics of the event time series. We particularly hypothesize that events in EHR-based multivariate event time series have dependencies with certain temporal structure and proper handling of various temporal dependency structures could enhance the predictability of a future event. Specifically, we focus on the following temporal structures of the EHR-based event time series:

1.4.2.1 Modeling Dependencies on Recent Events

In Chapter 3, we hypothesize that information on recently occurred events could provide strong predictability toward the next event occurrence. To properly model information from both recent and long-term past events, we develop a new event time series model based on the long-short-term-memory (LSTM) [77] that relies on two sources of information to predict future events. One source is derived from the set of recently observed clinical events. The other one is based on the hidden state space defined by the LSTM that aims to abstract past, more distant, patient information that is predictive of future events. In the context of Markov state models, the next state in our models and the transition to the next state is defined by a combination of the recent state (most recent events) and the hidden state summarizing more distant past events. We demonstrate the advantage of the proposed approach through extensive experiments on real-world EHR data and we show that our model outperforms multiple time series baselines in terms of the quality of event predictions.

1.4.2.2 Modeling Dependencies on Periodically Occurring Events

In Chapter 4, we hypothesize that (1) many events in the EHR-based multivariate event time series occur periodically and (2) proper modeling of the periodically occurring events could increase the predictability toward the next event prediction. For example, administrations of various medications occur with certain periodicity due to the nature of the medication administration dosage regime. Figure 7 shows the distribution of time gaps between two consecutive medication administration events for one of the medications with a typical period of 24 hours. One approach to modeling the periodicity of the time series is to rely on the hidden states of RNN/LSTM. However, when the number of the different periodic events in the EHR is large, it is not feasible to expect the model will be able to cover all periodic events using the same hidden state. To address this issue, we propose a novel yet simple mechanism to enhance the handling of periodic events and incorporate them into the prediction. Briefly, we equip an external memory that stores observed temporal characteristics of many periodic events and use them to derive a new periodicity-aware signal to further enhance event predictions. The external memory store gaps (time differences) were observed for pairs of two consecutive events of the same type (a) for all past patients and (b) for the current patient. At the time of the prediction, the proposed model calculates how much time has elapsed since the latest occurrence of the event of the same type, and based on the prediction window size and information stored in the memory of past event gaps, it predicts the probability of the signal to be repeated in the next prediction window. The proposed model achieves outperforming results compared to the baseline models as well as the model discussed in Chapter 3. Particularly for those events with notable periodic cycles in their occurrences, the proposed model shows remarkable performance gains.

1.4.2.3 Modeling Dependencies on Multiple Time-scales

In Chapter 5, we hypothesize that building predictive EHR representations is challenging due to the complexity of multivariate clinical event time series and their short and long-term dependencies to precursor events. We address this challenge by proposing a new neural memory module called Multi-scale Temporal Memory (MTM), linking events in a distant past with the current prediction time. Through a novel mechanism implemented in MTM, information about previous events on different time-scales is compiled and read on-the-fly for prediction through memory contents. We demonstrate the efficacy of MTM by combining it with different patient state summarization methods that cover different temporal aspects of patient states. We show that the combined approach is 4.6% more accurate than the best result among the baseline approaches, and it is 16% more accurate than prediction solely through hidden states of LSTM.

1.4.3 Research Goal 2: Learning Patient-specific Dynamic Representations

Another important challenge of learning good predictive models for clinical sequences is patient-specific variability. Depending on the underlying clinical condition specific to a patient combined with multiple different management options one can choose and apply in patient care, the event patterns may vary from patient to patient. Unfortunately, many modern event prediction models and assumptions incorporated into the training of such models may prevent one from accurately representing such a variability. Briefly, the parameters of neural temporal models are learned from many patients data through Stochastic Gradient Descent (SGD) and are shared across all types of patient sequences. Hence, the population-based models tend to average out patient-specific patterns and trajectories in the training sequences. Consequently, they are unable to predict all aspects of patient-specific dynamics of event sequences and their patterns accurately.

In this thesis, we want to address this patient-specific variability issue by **developing two novel adaptive event prediction frameworks that can adjust its prediction for individual patients**. Specifically, we want to first focus on modeling dynamics of heterogeneous multivariate event sequences by developing multiple sequential experts models that learn to adjust population model's prediction together. Then, we want to develop a more straightforward approach that adjusts the population model's prediction through patient-specific prediction models that are trained on each patient's own past event history. With these approaches, we expect to drastically improve the prediction performance of the events with low occurrences since they are typically observed in a few patients, and the personalized models can improve predictions of these events than clinical events observed in many patient sequences.

1.4.4 Hypotheses for Research Goal 2

1.4.4.1 Modeling Sequences with Adaptive Residual Mixture of Experts

In Chapter 6, we hypothesize that clinical event sequences in EHR are generated from a pool of heterogeneous patients where each patient typically has different types of complications. While average behaviors of clinical event sequences could be captured by a single model, the dynamics of heterogeneous event sequences could not be well captured by a single model. We address this challenge by proposing a specialized neural sequence model (RNN) based on the Mixture-of-Experts (MoE) architecture. The heterogeneity of various patient sequences is modeled through multiple experts that consist of Gated Recurrent Unit (GRU). Particularly, instead of directly training MoE from scratch, we augment MoE based on the prediction signal from the pre-trained base GRU model. In this way, the mixture of experts can provide flexible adaptation to the (limited) predictive power of the single GRU model.

1.4.4.2 Modeling Sequences with Personalized Online Adaptive Learning Framework

In Chapter 7, we hypothesize that (1) EHR-derived event sequences have patient-specific variability and (2) population-based models learned from such sequences may not accurately predict patient-specific dynamics of event sequences. Hence, we propose, develop, and study multiple new event sequence prediction models and methods that let us better adjust the prediction for individual patients and their specific conditions. The methods we develop pursue refinement of population-wide models to subpopulations, self-adaptation model, and a meta-level model switching that is able to adaptively select the model with the best chance to support the immediate prediction. These solutions extend RNN based multivariate sequence prediction to personalized clinical event sequence prediction.

1.5 Roadmap

We organize the thesis as follows: In Chapter 2, we define the multivariate event time series and review existing approaches to modeling the event time series from Markov models to modern autoregressive approaches based on neural networks. In Chapter 3 we present our work on predicting the next event from clinical event time series with recent context-aware LSTM model. In Chapter 4, we present work on predicting clinical event time series with recurrent event information through the specialized external information channel. In Chapter 5 we improve the prediction of future clinical events by linking past events in multiple time scales through a specialized external memory module. In Chapter 6, we present residual mixture of experts model that can enhance the one-model solution to adapt to the heterogeneity of the overall patient population and its subpopulations. Finally, in Chapter 7 we present novel personalized event time series prediction solutions that attempt to adjust the predictions for individual patients through an online adaptive model update mechanism and meta-switching mechanism.

2.0 Background

In this section, we first define the multivariate event time series, their representation and the prediction task considered in this thesis. After that, we review existing approaches relevant to multivariate time series modeling. Finally, we review existing methods and models for periodic signals and for the clinical event time series.

2.1 Multivariate Event Time Series

We define multivariate event time series by a time-stamped sequence of events $U = \{u_j\}_j$, where each event $u_j = [e_j, t_j]$ is represented by a pair of an event type e_j and its time t_j. We assume there are $|E|$ different event types defining the multivariate event time series. A univariate event time series would be defined by a single event type $|E|$=1.

The event time series with continuous time stamps can be directly modeled using point processes [82, 103, 178]. Examples of such processes are a Poisson process [98] or a Hawkes process [104, 180]. These models have been applied to various event sequence problems including clinical event prediction [122, 152]. However, these models are hard to optimize directly and the existing works only explore time series with a relatively small number of events. Because of these limitations, the event time series are often converted to discrete-time models (see Figure 8) where the original event time series are segmented using a window spanning some fixed period of time, and events within the window are considered to co-occur in the discretized time.

2.2 Segmentation (Discretization) of Event Time Series

We define the discrete-time event time series as follows:

- Discrete-time event time series $Y = \{y_i\}_i$ consist of a sequence of states y_i where $y_i \in$

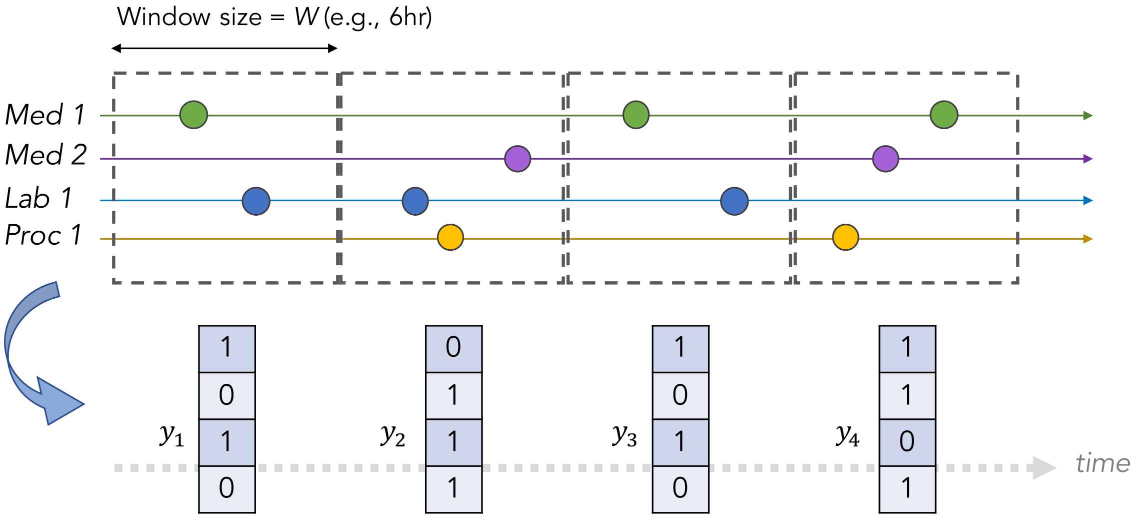


Figure 8: Overview of multivariate event time series processing. As seen in upper part of the figure, the original EHR-based time series data consists of event occurrences on continuous time. We discretize the time series with non-overlapping segmentation window and generate binary vector $y_i \in 0, 1^{|E|}$ that represents all event occurrences during the timings of the window.

$\{0,1\}^{|E|}$ is a binary vector that represents occurrences of events of different types at a discrete time step i, and $|E|$ denotes the total number of event types.

- Discrete-time event time series are generated from time-stamped multivariate event time series U through segmentation of event occurrences with a time window W as described in Figure 8.

In the following sections we assume we have data that consists of N discrete-time event time series: $D = \{Y_1, ..., Y_N\}$. Next, we briefly review existing modeling approaches for discrete-time event time series.

2.3 Markov Models

Markov models form a foundation of discrete-time series models. Given their simplicity and tractability, the majority of the event time series models are special cases of Markov models [137, 150]. Markov models represent an observed sequence of a random process over time as a sequence of states. The state is a categorical variable at a specific (discrete) time step. The *Markov property* assumes that the current state captures all necessary information relating to the future and past. In other words, the next state depends only on the most recent state, and is independent of past states:

$$P(y_T|y_{T-1}, y_{T-2}, ..., y_1) = P(y_T|y_{T-1}) \tag{1}$$

In this case, the joint distribution of an observed sequence is modeled as a chain of the conditional probabilities:

$$P(y_1, y_2, ..., y_T) = p(y_1) \prod_{i=2}^{T} P(y_i|y_{i-1}) \tag{2}$$

The conditional probability defining a transition is parameterized by a transition matrix $A \in \mathbb{R}^{|E| \times |E|}$:

$$A_{m,n} = P(y_i = n|y_{i-1} = m) \tag{3}$$

where $\sum_{n=1}^{|E|} A_{m,n} = 1$ for all m.

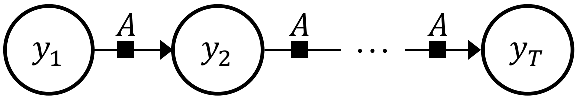


Figure 9: An illustration of a Markov model. The transition between observations y_is are defined by the transition matrix A in Equation (3)

The transition matrix A can be learned by the maximum likelihood estimation [176]. The standard Markov models assume all states of the time series are directly observed. However, the states of many real-world processes are not directly observable. One way to resolve the problem is to define the state in terms of a limited number of past observations or features defined on past observations [69, 70, 205] and another is to use the Markov models with hidden states.

2.3.1 Hidden Markov Models

The Hidden Markov models (HMM) [174, 197] introduce hidden states z_i of $d \times 1$ dimension. As shown in Figure 10, the observation y_i is modeled through the hidden state z_i and the emission table $B \in \mathbb{R}^{|E| \times d}$ with components: $B_{m,n} = P(y_i = n | z_i = m)$. Similar to the states in Markov models, the hidden state z_i is a categorical variable. The transition table A is used to update the hidden states and the emission table B is used to generate observations:

$$z_i = A \cdot z_{i-1} \quad y_i = B \cdot z_i \tag{4}$$

The parameters of the HMM (A, B) are trained with the Baum-Welch algorithm [18] which is a special case of Expectation-Maximization algorithm [45]. Given an observed sequence $y_1, ..., y_T$ and a trained HMM model (A, B), the most probable sequence of the hidden states $z_1, ..., z_T$ is computed by the Viterbi algorithm [211]. The prediction for next event y_{i+1} can be made straightforwardly, given the hidden state of the current time step z_i: $P(y_{i+1}|z_i) = B \cdot (A \cdot z_i)$.

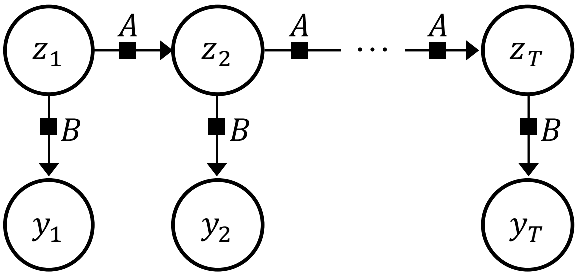


Figure 10: In a hidden Markov model, the observations $y_i; i = 1, \ldots, T$ are modeled through the the hidden states z_i through the emission matrix B. Dynamics of hidden states are modeled through the transition matrix A.

HMM has been shown to reach good performance in many applications such as stock price prediction [67], DNA sequence analysis [80], and time series clustering [194]. However, when applied to real-world time series, the classic HMM model has a drawback that its representational power is limited due to its discrete (categorical) hidden states and the transition of the hidden state is restricted between the discrete states. Linear dynamical systems (LDS) [56, 90] alleviate this issue by defining real-value hidden and observable states.

2.3.2 Linear Dynamical System

Linear dynamical system (LDS) [56, 90], also known as Kalman Filter, models time series Y with the real-valued hidden states. Specifically, as shown in Figure 11, LDS models the dynamics of the sequence as follows:

$$z_i = A \cdot z_{i-1} + \eta_i \quad y_i = B \cdot z_i + \zeta_i \tag{5}$$

where $A \in \mathbb{R}^{d \times d}$ and $B \in \mathbb{R}^{|E| \times d}$ are the transition and emission matrices, respectively. Unlike HMM, LDS explicitly models the stochastic component of the transition through the Gaussian noise $\eta_i \sim \mathcal{N}(\mathbf{0}, Q)$ where $\mathbf{0}$ is a $d \times 1$ dimension zero vector and $Q \in \mathbb{R}^{d \times d}$ is

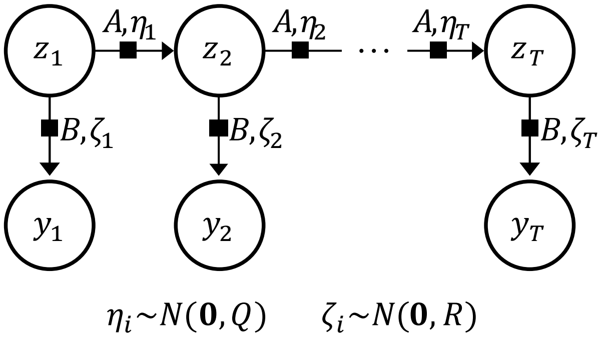


Figure 11: In linear dynamical system, an observation y_i is modeled through the hidden states z_i and the emission matrix B with the Gaussian noise $\eta_i \sim \mathcal{N}(\mathbf{0}, R)$. Dynamics of hidden states are modeled through the transition matrix A with a Gaussian noise $\zeta_i \sim \mathcal{N}(\mathbf{0}, Q)$.

a covariance matrix. The emission process that relates hidden states to observation also explicitly models the stochastic component through $\zeta_i \sim \mathcal{N}(\mathbf{0}, R)$ where $R \in \mathbb{R}^{d \times d}$. With regard to computing the value of the hidden state, LDS involves two inference tasks: *filtering* tries to compute the distribution of the hidden state z_i given the all previous and current observations $p(z_i|y_1 \dots y_i)$ and *smoothing* tries to compute the value of the distribution of z_n for a specific (intermediate) time step n given all observations from all past and all future ones $p(z_n|y_1 \dots y_T); 1 \leq n \leq T$. Details of the inference methods can be found in [27, 93, 132, 206]. The parameters of LDS $= \{A, B, Q, R\}$ can be learned through the Expectation Maximization (EM) algorithm [56] or spectral learning methods [47, 93, 126, 128, 206].

One issue with the hidden state in Markov models is that the dimensionality of their hidden state space is not known a priori. Various methods for hidden state space regularization, such as [126, 128] have been able to address this problem.

2.4 Neural-based Models for Event Time Series

Recent advances in neural architectures and their application to time series offer end-to-end learning frameworks that are often more flexible than standard time series models. In this section, we summarize neural-based methods for event time series processing: the recurrent neural network (RNN) and long short-term memory (LSTM).

2.4.1 Recurrent Neural Network

RNN is a type of neural network that models a sequence with the hidden state, similarly to HMM. But RNN is more flexible and efficient: given fixed input and target from data, RNN learns the intermediate association between them. Unlike HMM, the value of the hidden state of RNN is computed purely deterministically. Without any stochastic component, at each time step t, the hidden state h_t is computed given the previous time step's hidden states h_{t-1} and new information from the current time step's input y_t, with the following rule:

$$h_t = tanh(U \cdot y_t + W \cdot h_{t-1}) \tag{6}$$

where $tanh(\cdot)$ is a hyperbolic tangent and used as an activation function to help learn non-linearities. $U \in \mathbb{R}^{d \times |E|}$ and $W \in \mathbb{R}^{d \times d}$ are weight matrices. For brevity, we can also denote Equation (6) as follows:

$$h_t = \text{RNN}(y_t, h_{t-1}) \tag{7}$$

As shown in Figure 12, in RNN, the same weights are shared over time. Hence, no smoothing or filtering is required to compute the values of the hidden state. The prediction for the next event $\hat{y}_{t+1}$ is generated as follows:

$$\hat{y}_{t+1} = g(V \cdot h_t) \tag{8}$$

where $V \in \mathbb{R}^{d \times |E|}$ is an output layer weight matrix and $g(\cdot)$ is an output transformation function. $g(\cdot)$ can be any activation function and it needs to be selected to match the type of the target in data. For instance, if the target variable is a multi-class variable Softmax function is used. On the other hand, if the target is binary or is defined by a set of binary

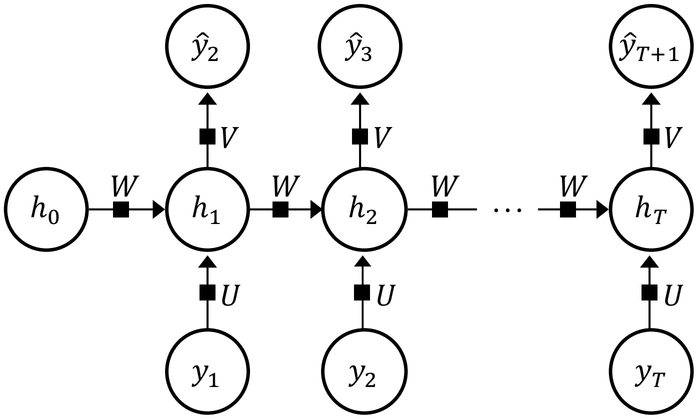


Figure 12: An illustration on architecture of recurrent neural network. At each time step t, the hidden state h_t is computed given the previous time step's hidden states h_{t-1} and new information from the current time step's input y_t.

variables a sigmoid function (such as the logistic function) is used. The parameters of RNN are learned through stochastic gradient descent (SGD). Loss is determined by cross-entropy function (multi-class) or binary cross-entropy function (multi-label), summed over all timesteps of each sequence as well as across all sequences [213].

Meanwhile, RNN is known to have limitations on learning and prediction with long sequences, referred as vanishing and exploding gradient [76]. Briefly, when the loss is propagated backward to update the weights, each weight receives an update proportional to the partial derivative of the loss. As Figure 13 shows, the gradient of $tanh(\cdot)$ is close to 0 at both ends. During the backpropagation, if a gradient is near to a small number at a timestep, it could make subsequent gradients also proximate to exponentially smaller numbers (as a gradient from later time step is multiplied to the previous one through the chain rule) and it can prevent weights to be updated properly. There are several potential solutions to mitigate this problem. One is to apply backpropagation on chunked sequence with a limited number of time steps (Truncated-BPTT) [198, 215]. Another is to add gates to produce paths where gradients can flow more constantly and longer without vanishing or exploding

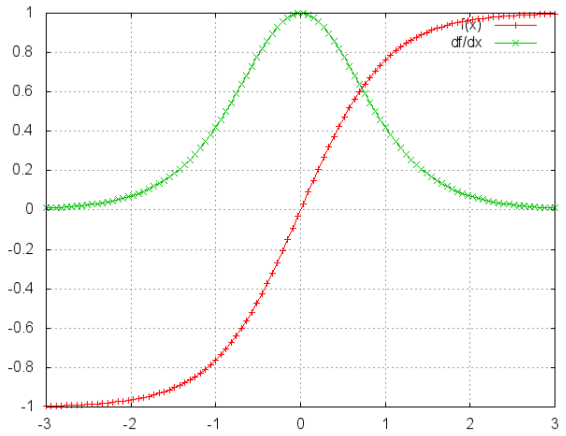


Figure 13: Ranges of the hyperbolic tangent (*tanh*) and its gradient. Source: `http://nn.readthedocs.org/en/rtd/transfer/`

(LSTM, GRU) [34, 77].

2.4.2 Long Short-Term Memory

LSTM effectively prevents the vanishing and exploding gradient problems with memory cell states and gates that control the information flow. Each gate is composed of linear transformation with a sigmoid activation function on h_{t-1} and y_t. In detail, the hidden states h_t and cell states C_t are updated as follows: first, LSTM updates the candidate for the new cell states $\tilde{C}_t$ as a function of h_{t-1} and y_t:

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, y_t] + b_c) \tag{9}$$

where $[,]$ represent concatenation of two vectors. Then, it computes forget f_t and input i_t gates which are used to determine how much content from the previous cell C_{t-1} will be erased and how much information of the new candidate cell states $\tilde{C}_t$ combine into the new cell state C_t respectively:

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, y_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, y_t] + b_i) \\ C_t &= f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \end{aligned} \tag{10}$$

Output hidden states h_t will be based on the cell state C_t with a filter from output gate o_t which decides which part of the cell state C_t will be in the output:

$$\begin{aligned} o_t &= \sigma(W_o \cdot [h_{t-1}, y_t] + b_o) \\ h_t &= o_t \otimes \tanh(C_t) \end{aligned} \tag{11}$$

where $\otimes$ denotes element-wise multiplication and $W_f, W_i, W_o \in \mathbb{R}^{|E|\times d}$ and $W_c \in \mathbb{R}^{d+|E|\times d}$. With these parameters ready, we can simply denote LSTM as a function of the previous hidden states h_{t-1} and current time-step's input y_t:

$$h_t = \mathrm{LSTM}(y_t, h_{t-1}) \tag{12}$$

The final prediction for next event $\hat{y}_{t+1}$ is computed the same way as RNN in Equation (8). The parameters are also trained through the same method used for RNN.

LSTM have been widely applied to many areas of prediction and modeling of sequence data such as time series [32, 66], vision [63], speech [61], and language [199] problems and many others.

2.4.3 Bidirectional RNN/LSTM

The methods covered in this section are based on an assumption that current state is dependent on past states. That is, we update and carry information in the hidden state h_t on *forward direction* from past to current (future) time ($t = 1, ..., T$). Meanwhile, for some application areas such as sentence classification or speech recognition in NLP, another information captured on *reverse direction* from future to past ($t = T, ..., 1$) would also be informative. Based on this motivation, Bidirectional RNN (BRNN) [186] extends a regular RNN by combining information from both directions, as shown in Figure 14. It independently updates two hidden states $h_t^{\rightarrow}, h_t^{\leftarrow}$ in opposing forward and backward directions.:

$$
\begin{aligned}
h_t^{\rightarrow} &= f(y_t, h_{t-1}^{\rightarrow}), \quad \text{for } t = 2, \ldots, T \\
h_t^{\leftarrow} &= f(y_t, h_{t+1}^{\leftarrow}), \quad \text{for } t = T-1, \ldots, 1
\end{aligned}
$$

where $f(\cdot)$ denotes an operation to update the hidden states such as Equation (7) for RNN and Equation (12) for LSTM. Two hidden states $h_t^{\rightarrow}, h_t^{\leftarrow}$ are updated independently and once they are computed, the final hidden state is computed as a concatenation of the two:

$$h_t = [h_t^{\rightarrow}, h_t^{\leftarrow}], \quad \text{for } t = 1, \ldots, T$$

BRNN has been used effectively in many NLP applications such as phoneme classification [62], text-to-speech synthesis [52], sequence tagging [79], and information extraction from free texts [85]. In the clinical domain, it is used to classify diagnosis code based on a sequence of clinical events [136]. Note that unlike how it is used in NLP, BRNN is not directly applicable to the prediction of the event time series as the future information is not available at the time of the prediction.

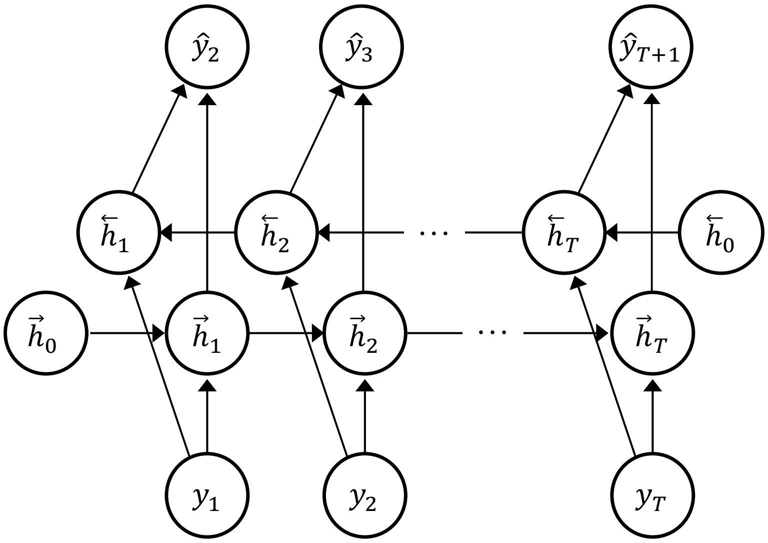


Figure 14: An illustration on architecture of bi-directional recurrent neural network. Note that, for brevity, the parameter matrices of the model are not represented in the figure.

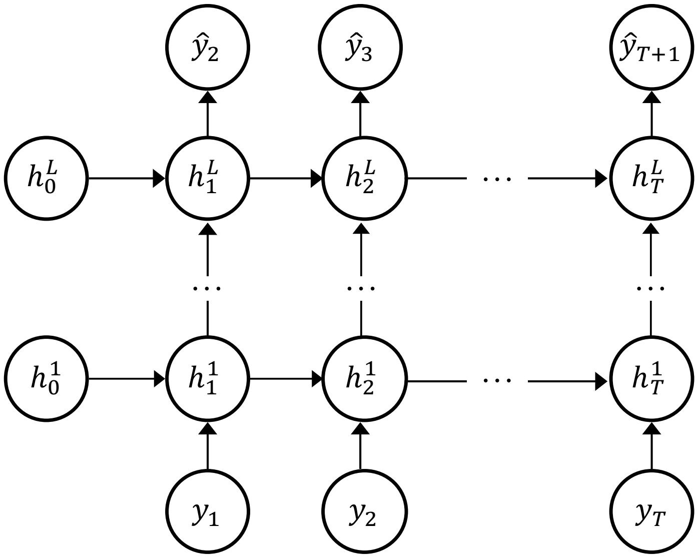


Figure 15: An illustration on architecture of hierarchical recurrent neural network

2.4.4 Hierarchical RNN/LSTM

In many cases, a sequence could have (latent) hierarchical structures. For example, a document consists of paragraphs and each paragraph consists of sentences. A sentence is a sequence of words and each word is a sequence of characters. A similar example is a video: a video is composed of a sequence of shots, and a shot is composed of a sequence of still frames.

The straightforward approach to modeling hierarchical structure on RNN/LSTM is to stack hidden states in several layers, as shown in Figure 15. We use $l = 1, \dots, L$ to denote an index for a layer in the hierarchical architecture. With the notations, we can represent the stacked hidden states as follows:

$$\begin{aligned} h_t^l &= f^l(h_{t-1}^l, h_t^{l-1}) \\ h_t^1 &= f^1(h_{t-1}^1, y_t) \end{aligned} \tag{13}$$

where f^l represents the operation that updates the hidden states at l-th layer. Based on the straightforward approach, several extensions have been developed to character level language modeling [41, 81], document modeling [118], video captioning [196], and video summarization [228].

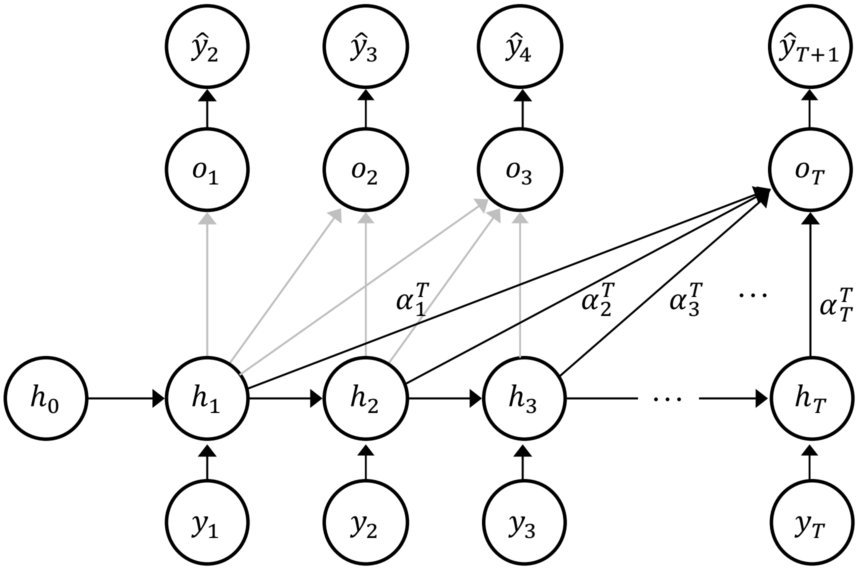


Figure 16: An illustration of attention mechanism. For brevity, the attention weights for an output o_T $(\alpha_1^T, \ldots, \alpha_T^T)$ are colored in black and other attention weights are colored in gray.

2.4.5 Attention Mechanism

When the length of a sequence is longer, it typically deters RNN/LSTM to learn dependencies between distant positions [75, 76]. Attention mechanism [5] tackles the challenge by using hidden states of *all* available time steps $h_1, \ldots, h_t$, instead of the last h_t. At current time step t, Attention mechanism generates an output o_t as a weighted sum of $h_1, \ldots, h_t$, as shown in Figure 16. Softmax is used to compute the attention weight α_i^t which measures relative importance of h_i among all available hidden states $h_1, \ldots, h_t$ to the output o_t. The weight is computed through as follows:

$$\alpha_i^t = \frac{\exp\big(\text{score}(h_i, q_t)\big)}{\sum_{j=1}^{t} \exp\big(\text{score}(h_k, q_t)\big)} \tag{14}$$

Typically, the previous time-step's output o_{t-1} is used for the query term q_t. In original paper [5], the score function is parameterized by a simple feed-forward neural network with tangent hyperbolic (tanh) activation:

$$\text{score}(h_i, q_t) = v_a \cdot \tanh(W_a \cdot [h_i, q_t]) \tag{15}$$

where v_a and W_a are weight vector and matrix. Then, we can compute the output o_t as a weighted sum of hidden states:

$$o_t = \sum_{i=1}^{t} \alpha_i^t \cdot h_i \tag{16}$$

For prediction, o_t is plugged into Equation (8) at the place where h_t is used: $\hat{y}_{t+1} = g(V \cdot o_t)$. Attention mechanism has been widely adopted in many machine translation and NLP tasks [29, 96, 135, 165, 168, 217].

2.4.6 Transformer (Self-Attention)

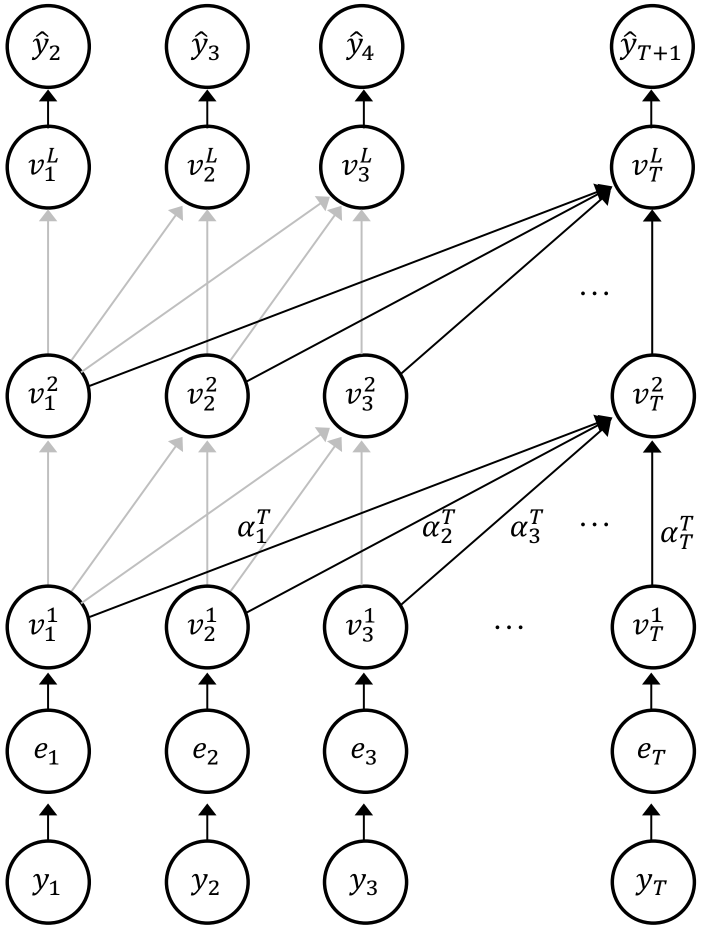


Figure 17: An illustration of Transformer architecture (self attention)

More recently, a new class of attention-based architecture, Transformer [209], removes the layer of RNN that computes hidden states recurrently. Instead, as shown in Figure 17, it models a sequence with multiple layers of self-attention mechanisms. Self-attention learns the internal (hidden) representation of each entry of a sequence as a weighted sum (attention mechanism) of all entities in the sequence in a previous layer.

More specifically, self-attention is constructed in a multi-layer architecture (l denotes index for a layer $l = 1, \ldots, L$) and each token y_i in an observed sequence $y_1, \ldots, y_T$ is represented as an internal state v_i^l. Briefly, v_i^l is computed as a weighed sum of all states

$v_1^{l-1}, \ldots, v_t^{l-1}$ in the previous layer:

$$v_i^l = \sum_{j=1}^{t} \alpha_{i,j} \cdot (v_j^{l-1})$$

$\alpha_{i,j}$ is a relative weight of tokens at time-step i and j and it is computed as a dot product of the two vectors v_i^{l-1} (itself in the previous layer), v_j^{l-1} (all input tokens in the previous layer $j = 1, \ldots, t$) followed by Softmax function:

$$\alpha_{i,j} = \frac{\exp\left(v_i^{l-1^\top} \cdot v_j^{l-1}\right)}{\sum_{m=1}^{t} \exp\left(v_i^{l-1^\top} \cdot v_m^{l-1}\right)}$$

The internal state v_j^1 at the lowest layer ($l = 1$) is computed by an embedding matrix $W_{\text{emb}} \in \mathbb{R}^{|E| \times d_{\text{emb}}}$: $v_i^1 = W_{\text{emb}} \cdot y_i$ where d_{emb} is the dimension of embedding.

By processing the internal states over multiple layers, the model is expected to learn representations that are better to relate different parts of the input compared to previous layers. The output is also computed as a weighted sum of the embeddings at the last layer L.

Besides, Transformer architecture features multi-head attention which enables different parts of a sequence to be attended different low-dimensional projection matrices: W_{query}^k, W_{key}^k, W_{value}^k (k denotes head index):

$$\begin{aligned} \alpha_{i,j}^k &= \frac{\exp\left((W_{\text{query}}^k \cdot v_i^{l-1})^\top \cdot (W_{\text{key}}^k \cdot v_j^{l-1})\right)}{\sum_{m=1}^{t} \exp\left((W_{\text{query}}^k \cdot v_i^{l-1})^\top \cdot (W_{\text{key}}^k \cdot v_m^{l-1})\right)} \\ v_i^l &= \sum_{k=1}^{K} \sum_{j=1}^{t} \alpha_{i,j}^k \cdot (W_{\text{value}}^k \cdot v_j^{l-1}) \end{aligned} \tag{17}$$

The motivation of using multiple heads is to allow the model to jointly attend to information that is differently represented in k subspaces.

2.5 Modeling Periodic Signals

Clinical event time series often come with temporal patterns defined by periodic events. In terms of modeling periodic signals, existing researches have traditionally focused on standard models defined by spectral decomposition of the signals using Fast Fourier Transformation (FFT) [23, 64, 74, 88, 164, 212, 12]. However, FFT is known to require sequential data with comparably high sampling rates [225] and due to this reason, FFT-based approaches may not fit with the modeling of clinical event time series data which consists of many sparsely occurring events.

Statistical parametric models are also used to model sparsely occurring temporal events. Based on hidden semi-Markov models, Kapoor and others [91] attempted to model repetitive music listening events. Trouleau and others [202] model video binge-watching behavior based on a Poisson mixture model with latent factors. Kurashima et al. [100] predicted everyday human actions from smart wearable devices with temporal point processes defined based on Weibull distributions. On the other hand, a simple histogram based approach is used to model inter-visit timing intervals for websites [1].

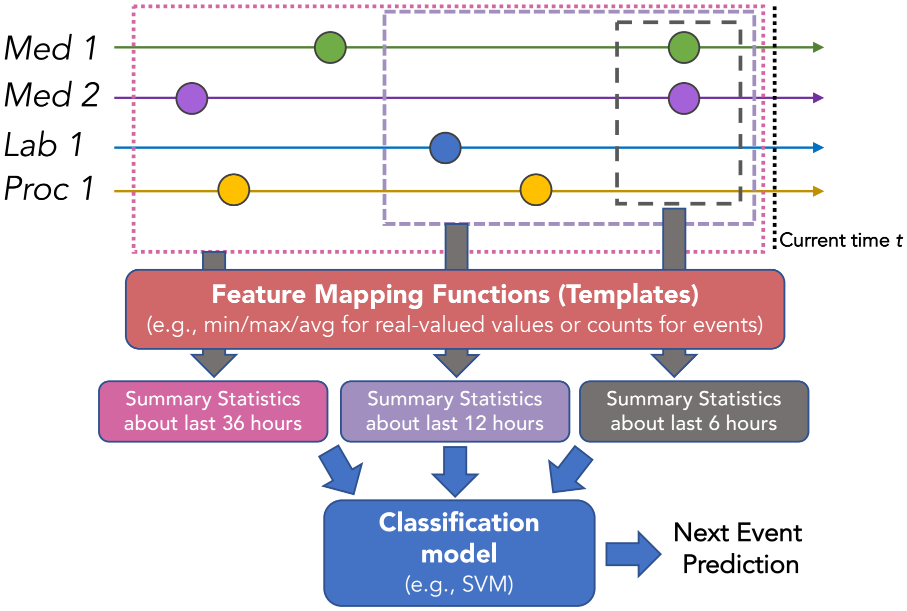

Med 1
Med 2
Lab 1
Proc 1
Current time *t*
Feature Mapping Functions (Templates)
(e.g., min/max/avg for real-valued values or counts for events)
Summary Statistics about last 36 hours
Summary Statistics about last 12 hours
Summary Statistics about last 6 hours
Classification model
(e.g., SVM)
Next Event Prediction

Figure 18: Classical EHR event prediction model based on feature templates and classification models method. Feature templates are defined as a set of feature functions that map clinical time series to fixed-size vectors (summary statistics) and their combinations over different aggregation time windows. The combinations of the summary statistics are fed to a classification model for next event prediction.

2.6 Clinical Event Time Series Modeling

In this section, we review major approaches for modeling clinical event-time series and their respective patient-state representation. This includes models based on both classical apriori defined featurizations of time series, as well as, recent automatic neural Deep Learning architectures.

2.6.1 Classical Models for EHR Event Prediction

Many traditional works on modeling clinical event time series are either based on standard classification models with a featurization scheme or based on probabilistic sequential models such as standard Markov models, HMM, or LDS. We discuss the details of both approaches below.

2.6.1.1 Approaches based on Feature Templates and Classification Models

Early works in EHR-derived event prediction tasks used special time series featurization procedures in combination with classification models [71, 205]. Briefly, an event prediction consisted of two steps: a time series featurization step and a classification step. The time series featurization analyzed the time series of observations prior to the time of the prediction and converted them to a fixed size feature vector. Standard classification models such as support vector machine, logistic regression, naive Bayes, or decision tree were then built directly on the new feature vectors.

The features representing time series in EHRs were often organized into expert defined temporal feature-template sets [69, 70, 71, 205]. A collection of the temporal feature templates when applied to many different time series in EHR are then used to dynamically (at any time) generate EHR time series summaries in terms of fixed-size feature vectors. As shown in the Figure 18, this method transforms complex multivariate clinical time series with either discrete and continuous-values into the fixed-sized vector representations. During the transformation, the method attempts to summarize dynamics of clinical time series, and it may incorporate features such as last glucose measurement, recent trend for the latest glucose level, nadir and apex values, trends from the baseline values, and the elapsed time since the last observation was made [71, 72]. Since this feature template method can represent complex dynamics of patient state, it is more suitable to build models for complex tasks such as, prediction of different types of clinical events [162, 204, 205] or outlier detection for clinical alerting [69, 70, 71]. We note that a number of variants and clones of this framework, making additional assumptions or restrictions about the conversion to fixed feature vectors exist. In general, the clones of the approach were successfully used to develop prediction models for onset of neonatal sepsis [144], depression [78], dementia[147], type2 diabetes [143], and for prediction of hospital readmissions [40, 166].

The temporal feature template method relies mostly on features developed by clinical experts. However, we note that features supporting the prediction can be also learned from data. One approach to do so relies on frequent pattern mining methods. Briefly, predictive pattern mining methods [13, 15, 14, 145] aim to identify the patterns, formed by logical combinations of observed inputs conditions that are able to predict with a high precision and support the target event or events. Temporal predictive pattern mining [9, 11, 16, 17] permits to relate the patterns in time, typically using temporal logic. In general, any predictive pattern identified by the pattern mining methods can be thought of as a special feature that helps to predict the target event. However, since the number of predictive patterns can be large only a carefully optimized subset is typically needed and used to support the prediction [9, 10, 11, 13].

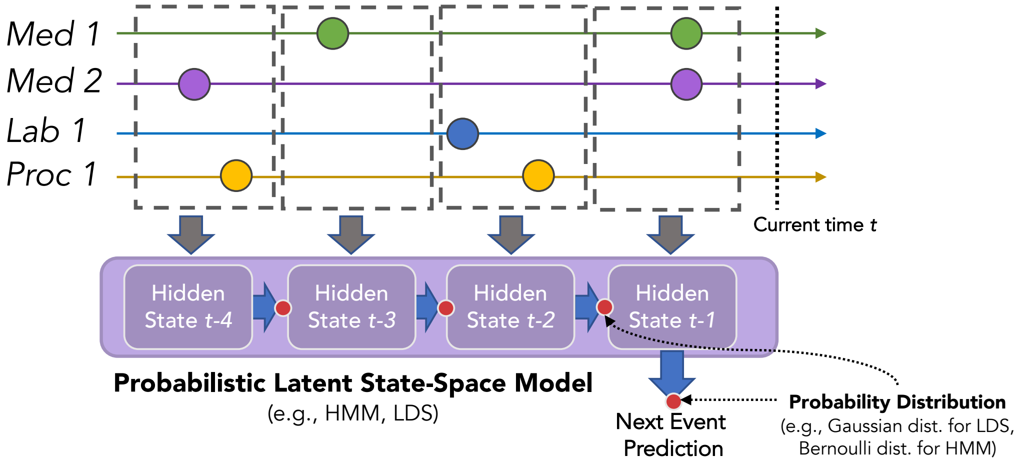


Figure 19: Clinical event prediction based on **probabilistic latent state-space models**, such as hidden Markov models, linear dynamical systems. The transition between hidden states and next event predictions are modeled through probability distributions.

2.6.1.2 Approaches based on Probabilistic Sequential Models

An alternative and more natural way of featurizing EHR-derived clinical event time series data is to build sequential probabilistic models that can automatically build the summary of past patient information from data, which is shown in Figure 19. This approach allows us to remove the expert defined featurization process presented above, and replace it with an automatic feature extraction and patient-state generation process.

There are many sequential models one can use to support EHR time series. Earlier sequential models use Markov models to model transitions of patient states which are represented by simple observable clinical variables (e.g., presence of certain clinical conditions or clinical outcomes). Although it was limited to represent complex patient state and its transitions, Markov models have been used in various clinical applications such as medical prognosis [20], simulating optimal timing of liver transplant [182], modeling patient compliance with medication administration regimens [216], modeling cardiovascular events for patients on antihypertensive treatment [187], estimation of survival probability for medullary thyroid cancer patients [49], and predicting replacement valve performances [43].

One issue with the standard Markov model is that it can model only the observable discrete patient state. Due to this, its representational power is limited for modeling transitions of complex patient states in EHR-derived sequence data. As introduced in Section 2.3.1 and Section 2.3.2, latent state-space models resolve this issue by introducing hidden state. Hidden Markov model (HMM) and Linear Dynamical System (LDS) are two notable models in this approach. Typically, the dimension of the hidden state is more compact than the observable patient state since one goal of latent state-space models is to find a compact representation that can encode important information about the past observation that is needed to predict future behavior in time series. Hence, these models are suitable to build prediction models for EHR-derived high-dimensional multivariate clinical event sequences. HMM and LDS have been used for a wide range of applications including septic shock prediction [57], early prediction of abnormal clinical events for chronic disease patients [55], mortality prediction [203], predicting patient-ventilator asynchronies [146], and prediction of heart failure decompensation events [163].

2.6.2 Neural (Deep Learning) Models for EHR Event Prediction

With recent advances in neural temporal models, the modeling of clinical event time series has adopted deep learning-based approaches to predict future clinical event occurrences given the history of longitudinal event sequences.

2.6.2.1 Approaches based on Word-to-Vector Models (Word2Vec)

The Word2Vec [22, 153] models are originally developed for computing distance between words in a low-dimensional projected space in natural language processing (NLP). For the training of the model, for example, in the Continuous Bag-of-Word (CBOW) algorithm [153], the objective (loss) is set to minimize the probability distributions of a center (target) word and its neighborhood (context) words for all words in documents of a training set. For the Skip-gram [153] based approach, the context and target are switched to each other. Once trained, the projection matrix, which is a learnable parameter of the model, is directly used to obtain a real-valued vector representation of the word.

For the clinical tasks, Word2Vec models have been adopted in a way that it gets a sequence of clinical events instead of words in the text data. Specifically, for the CBOW-based approach, recent events in a certain fixed-size history window are set as the context and an event that occurs shortly after the history window is set as a target. Word2Vec models are successfully applied to predict multivariate events on hospital visits [36] and diagnosis prediction [53, 114]. Specifically, [36] used Skipgram to predict clinical events that happened in neighboring (close past and future) visits given clinical event codes at the "current" visit. For architectural choice, the work uses a multi-layer structure first to merge embeddings of clinical events of a visit at a lower-layer and then to comprehend embeddings of demographic information at the next layer. In terms of clinical events, the work uses medical concepts such as diagnosis, medication, and procedure codes. [53] used a simple variant of Skipgram to predict diagnoses code at the next admission given sequence of context events at current admission. For clinical events, it uses lab tests and prescriptions. The model is trained in a way that the sum of context event embedding vectors should be close to an embedding of the diagnosis code that happens in the next admission. The same method is applied to each diagnosis code at the next admission. One drawback of the Word2Vec models is that they usually cannot fully model the sequential information, as they treat the events in the past equally when pooling (summing or averaging) past event embeddings. Besides, the size of the neighborhood (context) window is limited to a certain number of events (e.g., 20 or 40). Hence, those events that occur outside of the window cannot be used for modeling.

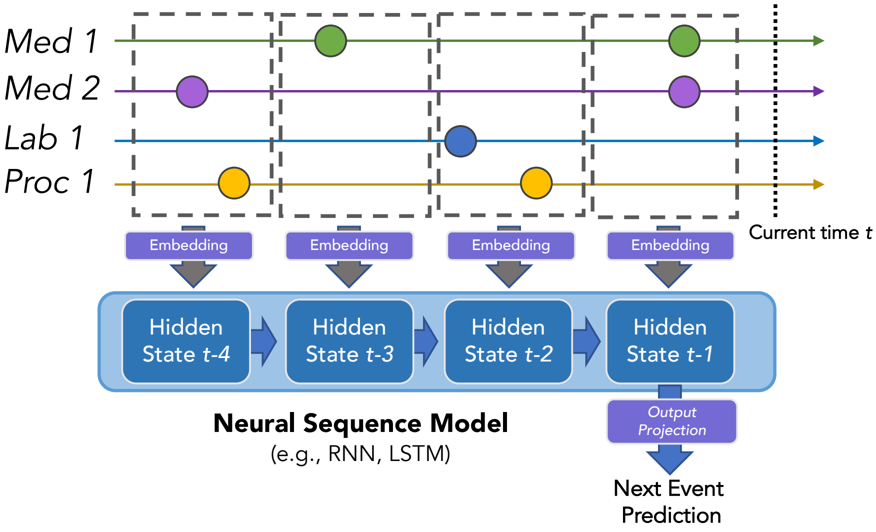


Figure 20: EHR event prediction based on **neural sequence models** such as RNN or LSTM. Compact representation of input is obtained by embeddings and transition between hidden states and output are modeled through non-linear functions such as Sigmoid (logit) function.

2.6.2.2 Approaches based on Neural Sequence Models (RNN/LSTM)

The sequence models based on RNN and LSTM [77, 213] resolve the problems by abstracting and carrying information from each step of the past through hidden states, as what is shown in Figure 20. As mentioned earlier in Section 2.4.1, at each time step, they recurrently update hidden states and for modeling of clinical event time series, the hidden states can be corresponding to a real-valued (latent) representation of patient states or management and action associate to the treatment of a patient. Another benefit of the RNN is that it can model all events in the entire sequence without a length-span limit, unlike Word2Vec. Hence, it has been deployed to various sequential clinical event modeling tasks.

Briefly, RNN and its variants have been successfully applied to many clinical event predictions such as medication prescriptions [6, 35], heart failure onset [39], readmission of chronic diseases [158], outcome of kidney transplantation [51], disease progression of diabetes [171, 173], mental health [169], and ICU mortality risk [224]. Specifically, [6] benchmarked performance of LSTM and Gated Recurrent Unit [34] models along with non-recurrent models such as random forests [28] on the task of predicting the next medication given a sequence of ICD-9 diagnosis codes. Also, [39] benchmarked GRU with non-sequential models such as SVM [200] and Multi-Layer Perceptron (MLP) for the task of predicting the onset of heart failure given events such as disease diagnosis, medication orders, and procedure orders that happened within a fixed observation window from longitudinal EHRs data. [35] used GRU to predict diagnosis and medication at a next visit given a sequence of diagnosis codes, medication codes or procedure codes in previous visits. [51] used RNN to predict the outcomes of kidney transplant operations, which were rejection of the kidney, loss of the kidney, or death of the patient give a sequence of medications, lab tests along with and demographic and static information about patients such as age, gender, blood type, weight, primary disease.

However, as mentioned in Section 2.4.5, RNN/LSTM could have difficulties in modeling long sequences due to the path where the loss is transferred. The loss (training objective function) is computed at the end of the sequence and the signal passed to parameters at each time step through the long-unrolled path of Back Propagation Through Time (BPTT) [214]. For RNN/LSTM, the maximum length of the sequence path is $O(n)$ (where n is the length of the sequence). The long path can hinder the proper transporting of loss signal and also the training of parameters could be affected [76, 209].

2.6.2.3 Approaches based on Attention Mechanism and Transformer

By creating a direct path $O(1)$ between the loss and any time-step in the unrolled path, the attention mechanism resolves the issue. Briefly, for the clinical sequence modeling, the attention mechanism has been applied to the treatment (medication) recommendation [227], prediction of sequential diagnoses and heart failure prediction [37], and prediction of in-hospital mortality, readmission rate, and length of stay [175] and in these works, attention-based approaches consistently show outperforming results over RNN/LSTM based models. Specifically, [227] developed a model that learns a mapping between a bag of diagnoses at a visit (input) and a bag of medications at the same visit (labels). The relationship between medications is modeled through attention mechanism and the relationship between medications and diseases is modeled through a RNN-based decoder, which sequentially predicts the most probable medication at a step. Interestingly, [37] used attention mechanisms to comprehend prior knowledge in medical ontology for sequence prediction. In detail, the authors developed a sequential diagnosis prediction model that predicts all diagnosis categories in the next visit. The model uses the attention mechanism over a tree-like structured knowledge graph (ICD-9 diagnoses code ontology) to compose a representation of a leaf diagnosis code as a weighted average of ancestor nodes. Leveraging prior knowledge in ontology to led better predictability compared to GRU-based baseline models. While [175] did not specifically predict clinical events such as medications or diagnosis codes, it benchmarked RNNs and the attention mechanism based methods on the prediction of important clinical outcome measures such as in-hospital mortality, readmission rate, and length of stay. They set the prediction to be made every 12 hours on a longitudinal large-scale EHR-based event time series, which consisted of about 2M patients from two major hospitals.

More recently, Transformer-based models with self-attention mechanism are used to represent patient states and dynamics [117, 177, 179, 195]. Specifically, [179] adapts bidirectional encoder representations from transformers (BERT) [46] for disease prediction task. [117] uses Transformer to predict the next mostly likely diseases in one's future visits. [177] built a Transformer-based model to predict incident heart failure. [195] demonstrated the competence of self-attention mechanism for predicting mortality and length of stay.

2.7 Personalized Clinical Machine Learning Models

The problem of fitting patient-related outcomes and decisions as close as possible to the target individual has been an essential topic of recent biomedical research and personalized medicine. We briefly list several approaches that build personalized machine learning models for clinical data in the following.

2.7.1 Subpopulation Models

One classic personalization approach identifies a small set of traits or features that help to define a subpopulation (patient subtype) the patient belongs to, builds a model for the subpopulation, and applies it if a patient from that subpopulation is encountered.

A straightforward way to define a subpopulation is to use initial clinical observations and demographics. For example, Afrose et al. [2] and Barda et al. [7] create patient subgroups with demographic traits such as race and age. They used the patient subpopulation to solve the data imbalance problem for underrepresented groups in predicting clinical outcomes such as mortality and length of stay. They first learn subpopulation-specific adjustment bias values for calibration purposes. Then, a model's classification outputs are adjusted based on the learned bias values.

Another approach to defining patient subpopulation is to use clustering methods. For complex clinical data with various types of features, this method has the advantage that it can reveal the latent (hidden) structure and relationship regardless of the complexity of the data representation. In addition, a clustering method can be used for any data representation where the distance metric (or equivalently similarity measure) between data points can be defined.

Many earlier works on this approach focused on clustering static patient representation such as demographics and symptoms of disease [105, 116, 170, 207]. More recent work focus on clustering longitudinal patient representation such as trajectories of biomarkers of kidney function [134], opioid usage [155], or lab test orders [190]. Since this approach considers dynamic changes of clinical features in the data, the discovered patient clusters provide a valuable opportunity for clinical data analysis, such as understanding disease progression [134, 190] or developing more accurate prediction models [155]. For clustering, many earlier works directly use K-means, DBSCAN, or hierarchical clustering algorithms on the top of the features [48, 134, 155, 190], and recent works use deep learning based methods to obtain more compact feature representation over the complex clinical data [19, 226].

2.7.2 Patient-specific Models

A more flexible approach to personalized clinical models is to develop patient-specific models that can identify the subpopulation of patients relevant to the target patient by using a patient similarity measure and then build and apply the patient-specific model online whenever the prediction is needed [54, 181, 210].

Another approach to developing patient-specific models is to use probabilistic sequential latent variable models such as Gaussian Process [184] and Hidden Markov Model [188]. These models have a certain probabilistic form, such as Gaussian distribution for real-valued observations. The parameters for the probability distribution (e.g., mean and variance for Gaussian distribution) are learned during the training process. To build a *personalized* probabilistic latent variable model, patient-specific terms are added to the probability distribution parameters. This approach has shown good performance for predicting lab test value (trajectory) of lung disease patients [184] and future complications of Parkinson's disease [188].

2.7.3 Online Adaptation Methods

However, in many sequential prediction scenarios, the models that are applied to the same patient more than once create an opportunity to adapt and improve the prediction from its past experiences and predictions. This online adaptation lets one improve the patient-specific models and their prediction in time gradually. The standard statistical approach can implement the adaptation process using the Bayesian framework where population-based parameter priors combined with the history of observations and outcomes for the target patient are used to define parameter posteriors [24]. Alternative approaches for online adaptation developed in literature use simpler residual models [127] that learn the difference (residuals) between the past predictions made by population models and observed outcomes on the current patient. Liu and Hauskrecht [127] learn these patient-specific residual models for continuous-valued clinical time series and achieve better forecasting performance.

2.7.4 Online Switching Methods

The online switching (selection) method is a complementary approach that has been used to increase the prediction performance of online personalization models by allowing multiple (candidate) models to be used together [120, 189]. At each time in a sequential process, a switching decision is made based on the recent prediction performance of each candidate model. For example, for continuous-valued clinical time series prediction, Liu and Hauskrecht [129] have a pool of population and patient-specific time series models, and at any point in time, the switching method selects the best performing model.

3.0 Modeling Clinical Event Time Series with Recent Temporal Mechanisms

3.1 Introduction

As mentioned in Section 1.4, modeling EHR-based event time series imposes several challenges due to heterogeneous temporal dependencies. Particularly in this chapter, we propose to tackle the challenge of different temporal characteristics in the time series by developing a novel autoregressive time series model that compiles multivariate event time series using multiple temporal mechanisms that cover different temporal characteristics of EHR-based event time series: The patient information from longer-term distant past is abstracted through hidden states of the **neural abstraction module** that is based on Long Short-term Memory (LSTM) [77]. The recent information on the patient state is compiled by **recent context module** that projects the recent event information into discriminative space.

To evaluate our model, we use the real-world clinical data derived from EHRs of critical care patients in the MIMIC-III database [89]. The clinical events considered in this work correspond to multiple types of events, such as medication administration events, lab test result events, physiological result events, and procedure events. These are combined in a dynamically changing environment typical of intensive care units (ICUs) with patients suffering from severe life-threatening conditions. Through rigorous evaluations on MIMIC-III data we show that our model outperforms multiple baseline models in terms of the quality of event predictions. To provide further insights into its benefit and prediction performance, we also split the results with respect to different types of clinical events considered (medication, lab, procedure, and physiological events), as well as, based on their recurrence patterns, again showing the superior performance of our model.

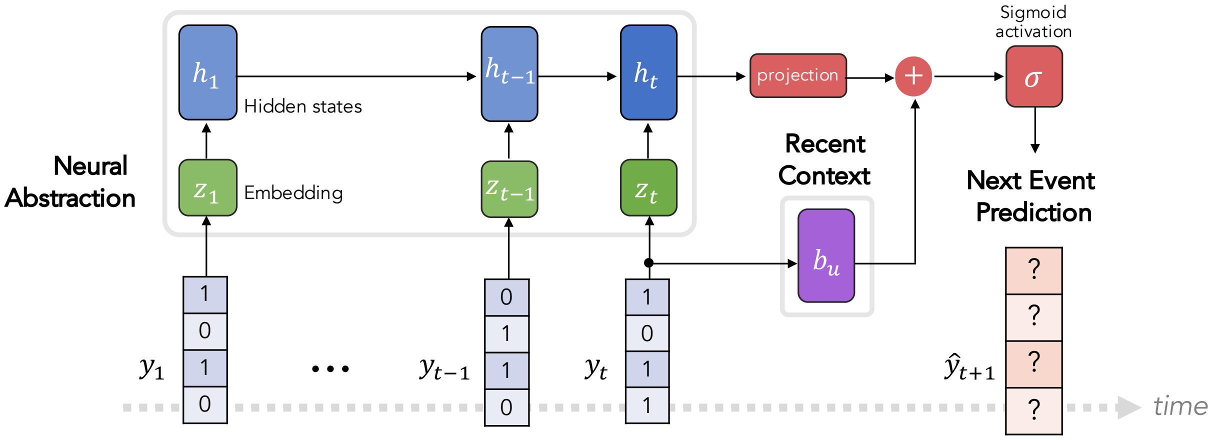


Figure 21: Architecture of the proposed model. Multivariate event time series $(y_1, \ldots, y_t)$ are processed by two information channels: recent event information is processed through the recent context module and history information is processed through the neural abstraction module.

3.2 Methodology

As Figure 21 shows, different temporal aspects of information from the multivariate event time series $(y_1, \ldots, y_t)$ are processed through different mechanisms. At a high level, information from distance past is abstracted and carried through the hidden states of the LSTM-based neural abstraction module. Information from a recent context is processed through the context module. The model combines two channels of information and outputs the probabilities of multivariate event occurrences of the next time step. In the following, we describe each module in detail.

3.2.1 Neural Abstraction Module

LSTM models are being successfully used to model time series with the help of hidden state vector, allowing one to summarize in the hidden state information from a more distant past. At a glance, at each time step of a sequence, LSTM gets current (event) input and updates its hidden states. The hidden state then generates signals for the next hidden state, as well as, predictions for the occurrence of events in the next time-step.

In detail, at each time step t, events in the input sequence represented as multi-hot vector y_t are processed and mapped to a real-valued vector z_t through embedding matrix W^{emb}: $z_t = W^{(emb)} \cdot y_t$. Then, given the processed input z_t and previous hidden states h_{t-1}, LSTM updates hidden states h_t through the update rules defined in Equation (9)-Equation (11):

$$h_t = \text{LSTM}(z_t, h_{t-1}) \tag{18}$$

3.2.2 Recent Context Module

When properly trained, the hidden state in the LSTM module can be sufficient to represent and model future behaviors of event time series by abstracting dependencies of past and future events. However, to be trained properly, LSTM (or any deep-learning based models) requires large amounts of training instances. In the clinical domain, obtaining large amounts of clinical cases (e.g., rarely ordered medication or lab tests) is hard in general. This constraint may deter us to train LSTM for predicting rare clinical cases. Meanwhile, for certain clinical event categories such as medications, the future occurrence of an event may highly depend only on the most recent events, and not distant past. Hence incorporating this information through the hidden state of LSTM does not make much sense. To address the above issues, we propose to distinguish and model two sources of information from past event sequence: (1) the abstracted information of past event sequence through hidden states of LSTM representing more distant past and (2) the specific information about event occurrences in a very recent context window. The recent context module serves to capture and process the recent event information. Briefly, the recent event at the current time step t is in multi-hot vector y_t and it is incorporated into the model through a linear transformation to model:

$$b_u = W_s \cdot y_t + b_s \tag{19}$$

b_s can be seen as additional bias term that reflects recent event occurrence information.

3.2.3 Combining Predictive Signals

Final prediction for event occurrence is made as follows:

$$\sigma(W_{out} \cdot h_t + b_{out} + b_u) \tag{20}$$

where $W_{out} \in \mathbb{R}^{|E| \times (h)}$ and $b_{out} \in \mathbb{R}^{|E|}$ are parameters of the linear transformation of the vector combining the two signals. The proposed predictor combines information on distant past from LSTM's hidden states and the recent state (most recent events) information as an additional *recent bias* term. The addition of the recent bias can be seen as adjusting information from LSTM's hidden states with other information from recent event occurrences.

3.2.4 Parameter Learning

The parameters of the model are learned by backpropagation through time (BPTT) [214] with an adaptive stochastic gradient descent based optimizer (Adam) [97]. For loss function $\mathcal{L}$, we use binary cross entropy between the prediction vector $\hat{y}_t$ and the true event occurrence vector y_t over all sequences in the training set and $\mathbf{1}$ denotes a vector filled with 1s:

$$\mathcal{L} = \sum_t -[y_t \cdot \log \hat{y}_t + (\mathbf{1} - y_t) \cdot \log(\mathbf{1} - \hat{y}_t)]$$

3.3 Experimental Evaluation

In this section, we evaluate the performance of our new autoregressive model on MIMIC III data [89] and compare it with alternative baselines.

3.3.1 Clinical Data

We test the proposed model on MIMIC-III, a clinical database generated from real-world EHRs of intensive care unit patients [89]. We extract 5137 patients from the database by applying the following selection criteria: (1) adult patients, with age is between 19 and 99 (2) patients with length of ICU is between 48 and 480 hours, and (3) patients with records represented in the Meta Vision, one of the systems used to generate MIMIC-III dataset. Except for these criteria, we do not filter out any patient in order to test our model across the general patient pool regardless of disease, symptoms, or conditions. We split the patients into the training and test sets with a ratio of 8:2.

For the sake of the robust experimental evaluation, we build 10 different train-test data splits by randomly shuffling the patients before splitting. We report averages over these 10 different splits.

3.3.2 Feature Preparation

We generate discrete-time event time series by segmenting all EHR sequences with three window sizes (W=6,12,24 hours). As mentioned in Section 2.1, at each step of a window segment, the input y_t is formed by aggregating all types of events in the window as a multi-hot vector and the prediction target y_{t+1} is formed as a multi-hot vector of events that occurred in the next window segment. EHR contains thousands of different clinical event types. For efficient modeling we use clinical events that are representative of patient conditions and clinical actions. With this regard, we use four clinical event categories: medication administration events, lab results events, procedure events, and physiological result events. Recent studies in clinical event prediction for EHR data show that using occurrence information (presence/absence) of laboratory tests is more informative than using the measured values of laboratory tests [3, 191]. Hence, for the lab test and physiological results events, we use occurrence information of each event instead of the values of the observation. For medication, lab, and procedure event categories, we filter out those events observed in less than 500 different patients. For physiological events, we select 16 important event types with the help of a critical care physician.

Further, for each of 10 splits, we filter out those events that are not observed in both train and test sets. The number of resulting events ($|E|$) is 282. The Table 1 shows relevant data statistics collected from the train set.

Category	Medication	Procedure	Lab test	Physio signal
Cardinality	64	44	155	19
Num. of occurrences	59K	53K	308K	181K
Proportion of positive label	5.8%	7.6%	12.7%	60.9%

Table 1: Clinical data statistics by event categories (W=6). Proportion of positive label is computed as of the frequency of the event occurrences in the segmented 6-hour time windows.

3.3.3 Baseline Models

We compare our model with multiple baseline models that are able to predict events for multivariate event time series given their previous history. The baselines are:

- **Logistic regression based on the recent history information (LR-Recent)** predicts the next event occurrence y_{t+1} using the current events y_t. The model is defined by a linear transformation with the sigmoid output function: $\hat{y}_{t+1} = \sigma(W_{lr} \cdot y_t + b_{lr}), W_{lr} \in \mathbb{R}^{|E| \times |E|}$.
- **Logistic regression based on the full history (LR-Binary)**: aggregates all event occurrences from the complete past event sequence and represents them as a binary vector. The vector is then projected to the prediction of y_{t+1} by using the same parameterization as the above model.
- **Logistic regression based on the hidden states from LSTM (HS)**: predicts y_{t+1} based on the hidden states of the LSTM in Equation (18). Linear transformation with sigmoid activation function is used similarly to the above models.
- **Reverse-Time Attention Mechanism (RETAIN)**: RETAIN is a representative work on using attention mechanism to summarize clinical event sequences, proposed by Choi et al. [38]. It uses two attention mechanisms to comprehend the history of GRU-based hidden states in reverse-time order. For multi-label output, we use a sigmoid function at the output layer.
- **Logistic regression based on convolutional neural network (CNN)**: This model uses CNN to build predictive features summarizing the event history of patients. Fol-

lowing Nguyen et al. [159], we implement this CNN-based model with a 1-dimensional convolution kernel followed by ReLU activation and max-pooling operation. To give more flexibility to the convolution operation, we use multiple kernels with different sizes (2,4,8) and features from these kernels are merged at a fully-connected (FC) layer.

The proposed model that combines the two sources of information (Hidden States from LSTM and Recent Context State) is referred to as **HS-RC**.

3.3.4 Evaluation Metrics

We evaluate the quality of predictions by calculating the area under the precision-recall curve (AUPRC). AUPRC is known for presenting a more accurate profile on performances of models under a highly imbalanced dataset [183]. Due to the nature of EHR-derived time series data, our dataset is highly skewed to negative examples as shown in Table 1.

The reported AUPRC values (for the different methods) are averaged over all target events and over test sets defined by 10 different train/test splits.

3.3.5 Implementation Detail

For the experiments, we use embedding size 64 and fixed learning rate=0.005 and mini-batch size=256. The size of the LSTM's hidden states is determined by internal cross-validation set with ranges of $(64, 128, 256, 512)$. To prevent over-fitting, L_2 weight decay regularization is applied to all models and the weight is also determined by the internal cross-validation.

3.3.6 Experiment Results

Figure 22 summarizes prediction results for all event types for three window sizes ($W = 6, 12, 24$) by averaging AUPRC obtained on our model and baselines. We observe that our model, HS-RC, dominates in smaller window segments W=$6, 12$ and is no worse than its component models in larger window segments W=24. Also, note that AUPRC results for larger window sizes are higher. This is expected since segmentations based on larger

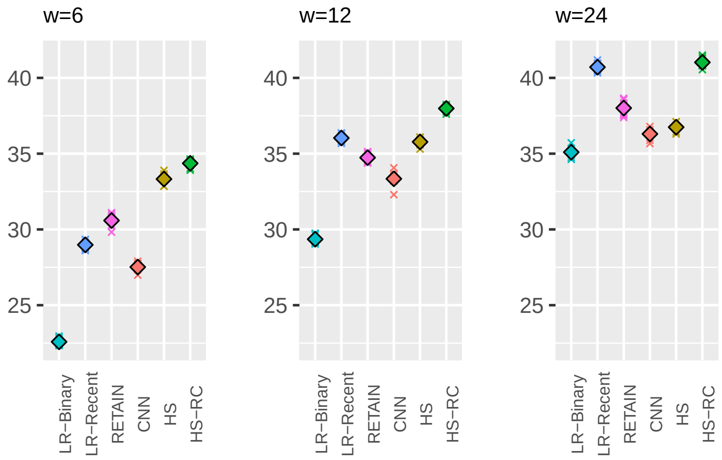


Figure 22: Overall prediction results. The results show average test AUPRCs over all events and 10 different random train-test splits.

window sizes lead to higher priors for the occurrence of the events. When the window size is W=24, the performance of the LR-Recent model approaches close to HS-RC. On the other hand, the performance of the LR-Recent on smaller window sizes deteriorates rapidly. This suggests that our model, HS-RC, learns to pick up important predictive signals between recent context and hidden states (that could get information from long past), depending on the best predictive information given current window segmentation setting. It also suggests that most of the important information for predicting future clinical events comes from the recent 24 hours. This finding can be also partly explained by the fact that many events (such as drug administrations or lab orders) are repeated every 24-hours, hence once they are observed they are most likely to occur also in the next time window. In terms of pure HS model, the difference from HS-RC model is more visible across all window sizes, but HS contributes to HS-RC predictions visibly more for small window size (W=6), which is in line with the observed reduced benefit of LR-Recent model for that window size.

3.3.6.1 Analysis of Results based on Event Categories

To analyze the experiment results further, we next break the evaluation results down by inspecting the predictive performances of the models for the four different event categories: medication events, lab events, physiological events, and procedure events. The results are shown in Figure 23. We can see that on medication administration prediction, the predictive signal from the recent context (LR-Recent) is much higher than the signal from hidden states of LSTM (HS) for larger window sizes ($W = 12, 24$). But at shorter window sizes ($W = 6$) we see that hidden states of LSTM show better predictability than recent context. This could be caused by fine granular and much longer sequences resulting from shorter window sizes ($W = 6$). Under this situation, long-term information brought by hidden states is more valuable to make the prediction for the next medication event. One important fact is that regardless of different window sizes, our model (HS-RC) is able to fuse the information from both channels and shows the best predictability.

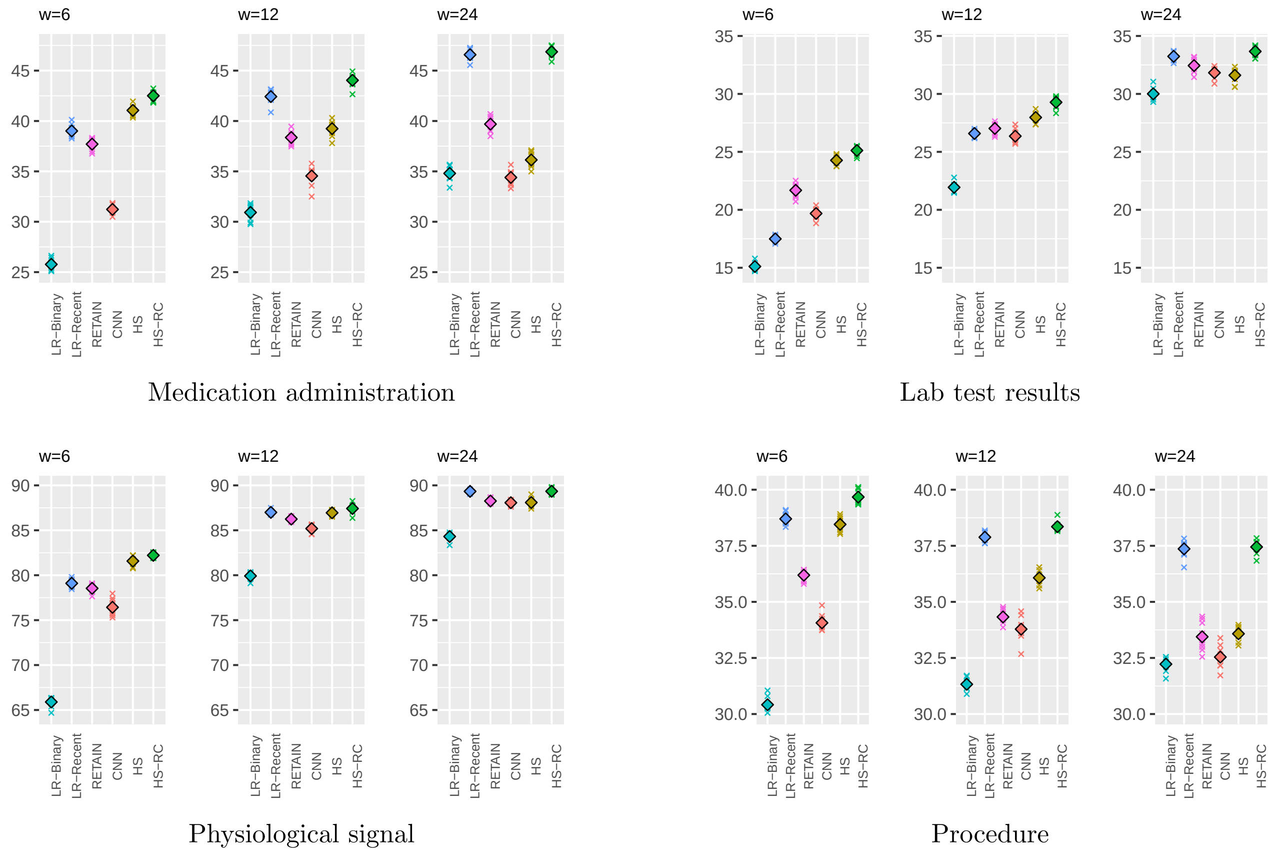
For lab test results prediction, our model also dominates across all time windows. For $W = 24$ time window, as we could expect, the more predictive signal is from recent context (LR-Recent) than hidden states of LSTM (HS). But at shorter time windows ($W = 6, 12$), the contribution from hidden states seems much higher than ones from recent context.

On physiological signal prediction, the performance gap between recent context, hidden states, and our model is overall smaller than other event categories. At the same time, the overall predictability of all models is higher than other event categories. This could be the case that many event types in the physiological signal are ones that are from bedside monitoring devices and they have more regular occurrence patterns from a repeated collection of the signals.

On procedure event prediction, recent context (LR-Recent) shows higher predictability than hidden states (HS) across all time windows. Interestingly, the performance of hidden states from LSTM deteriorates quickly as we have larger time windows. It might be a case that regardless of the window sizes, recently occurred events bring the most important information to predict the next procedure event. At the same time, as we have larger time windows (e.g., $W = 24$), contents in hidden states from LSTM are packed with non-essential signals for predicting procedure events. Since our model training scheme and prediction task is to predict all different types of clinical events at the same time, this could be the reason, and this also shows the competency of our proposed model that is capable of referring information signals from channels that cover different time ranges: one from recently occurred contextual events and another from long-term past events in history.

3.4 Summary

In this chapter, we showed the importance of processing multivariate event time series with different temporal mechanisms that aims to process different temporal aspects of the time series. Information related to distant past is modeled through the hidden state space defined by LSTM and information on recently observed clinical events is modeled through discriminative projections. We show that our model equipped with the two information channels leads to improved prediction performance compared to the baselines by learning to pick up information from the best predictive source.



Medication administration

Lab test results

Physiological signal

Procedure

Figure 23: Prediction results by the event type category

4.0 Modeling Clinical Event Time Series with Recurrent Temporal Mechanisms

4.1 Introduction

One important characteristic of EHR-based event time series is that they are often periodic and events are repeated regularly in time. For example, some medications are administrated with certain periodicity due to the nature of the medication's dosage regime. Since periodic or quasi-periodic events are quite frequent in EHRs, in order to define highly accurate event prediction model the periodicity of the events and their occurrences need to be adequately modeled. One approach to modeling the periodicity of the time series is to rely on the hidden states of RNN/LSTM. However, when the number of the different periodic events in the EHR is large, it is not feasible to expect the model will be able to cover all periodic events using the same hidden state. To address the issue, we propose a novel yet simple mechanism to enhance the handling of periodic events and incorporate them into the prediction. The patient information from repeatedly occurring events is modeled through **periodicity module** that consists of an external memory that stores observed temporal characteristics of many periodic events and uses them to derive a new periodicity-aware signal to further enhance event predictions. At the time of the prediction, the module calculates how much time has elapsed since the latest occurrence of the event of the same type, and based on the prediction window size and information stored in the memory of past event gaps, it predicts the probability of the signal to be repeated in the next prediction window. The main advantage of the approach is that it is modular and can be used in combination with other patient history summarization mechanisms. In the chapter, we model and predict future events of the multivariate clinical time series based on *combination* of the periodicity module with the modules introduced in Chapter 3, neural abstraction module and recent context module. With the combination of the modules, our model is capable of summarizing and utilizing different aspects of complex clinical event time series toward accurate prediction of future event occurrence.

4.2 Methodology: Periodicity Memory Module

Many events in the EHR-based multivariate event time series occur periodically. For example, administrations of various medications occur with certain periodicity due to the nature of the medication administration dosage regime. Figure 27 shows the distribution of time gaps between two consecutive administration events for one of the medications with a typical period of 12 hours.

One approach to modeling the periodicity of the time series is to rely on the hidden states of RNN/LSTM. Briefly, if the RNN is properly trained, it could figure out sufficient statistics and counting processes needed to drive periodic signals. However, when the number of the different periodic events in the EHR is large, it is not feasible to expect the model will be able to cover all periodic events using the same hidden state (of limited size). To prevent this from happening, we propose a simple mechanism to enhance the handling of periodic events and incorporate them into the periodicity module. Briefly, the new module relies on a memory that stores observed temporal characteristics of many periodic events and uses them to derive a new periodicity-aware signal to further enhance event predictions, and this at any time, and for any prediction window size.

In a nutshell, our module uses memory that stores gaps (time differences) observed for pairs of two consecutive events of the same type (a) for all past patients and (b) for the current patient. At the time of the prediction for the current event time series, the module calculates how much time has elapsed since the latest occurrence of the event of the same type, and based on the prediction window size and information stored in the memory of past event gaps, it predicts the probability of the signal to be repeated in the next prediction window. As noted earlier, two different sources of information are used: (a) event gaps for the current patient and (b) compiled event gap distributions obtained from time series of past patients in the training set. We describe the event prediction mechanisms in more detail in the next subsections.

4.2.1 Event Prediction Based on Recent Event Gap for the Current Patient

The periodicity module models periodicity of individual patient's event stream and utilizes it for predicting the next occurrence of each event type. To predict the future occurrence of event $e \in E$ for the current patients, we use two periodicity-related statistics:

- **Recent interval** (ζ), that is a time period between the two most recent occurrences of the event e in the current event stream:

$$\zeta_t^e = \tau_t^e - \tau_{t-1}^e$$

  where τ_t^e and τ_{t-1}^e are timings of the two most recent occurrences of the event e in the current event stream (that is, events closest to current time t).

- **Elapsed time** (ϵ) that is the time elapsed from the last occurrence of the event e in the current stream:

$$\epsilon_t^e = t - \tau_t^e$$

  where t denotes the current time.

With the above two statistics, the model outputs patient-specific periodicity-based prediction p_t^e for event e and the prediction window of size W: as

$$p_t^e = \begin{cases} 1 & \text{if } \epsilon_t^e < \zeta_t^e < \epsilon_t^e + W \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$

One drawback of this approach is that it cannot make predictions until it observes the first two occurrences of events (τ_1 and τ_2). In addition, the recent interval statistic keeps only recently observed time gap between the two consecutive events for the current patient and hence its predictions may become inaccurate. To address this issue, we rely on event gap statistics and their distribution as obtained from the training patient set.

4.2.2 Event Prediction Based on Event Gap Distribution of Past Patients

The probability distribution of event gaps (time differences between two consecutive events) for each individual event type can be compiled from across all patients in the training

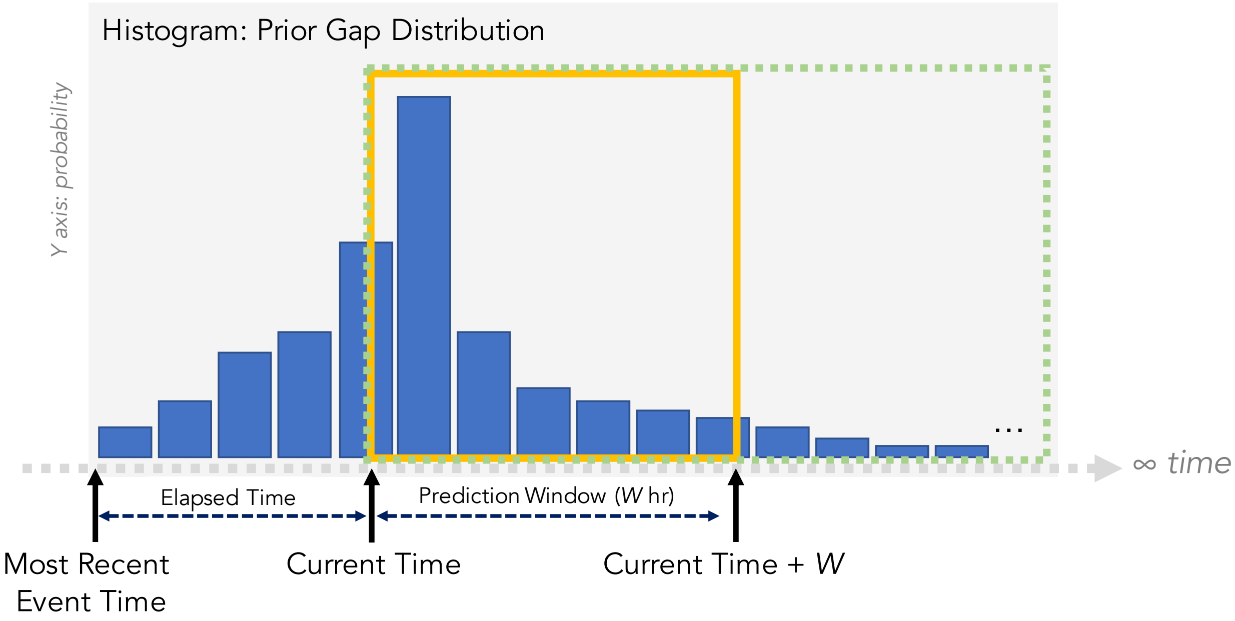


Figure 24: Event gap distribution and calculation of the probability of future event occurrence. Event gap distribution is formed by summarizing event gaps observed in time series of past patients and compiling them into a (normalized counts) histogram. The event prediction for the current time t and prediction window W is made by calculating the histogram probability mass defined by the time elapsed since the last event and the size of the future prediction window (sum of probabilities inside yellow solid line) and by normalizing it with the remaining probability mass defined by current time and onward time (sum of probabilities inside green dotted line).

set and represented (non-parametrically) in a histogram structure. Figure 24 illustrates such a histogram.

To predict the probability of occurrence of the next event e in the prediction window W for the current patient's time series, the histogram structure, the time elapsed since the most recently observed event ϵ_t^e, and size of the prediction window W are considered. To do so, we define a function $mass(t_a, t_b)$ that returns the probability mass between two time points $[t_a, t_b]$ (such that $t_a < t_b$) when projected on the histogram. Then, the signal r_t^e predicting the probability of the next event occurrence within the next time window based on the histogram probabilities is defined as follows:

$$r_t^e = \frac{mass(\epsilon_t^e, \epsilon_t^e + W)}{mass(\epsilon_t^e, \infty)} \tag{22}$$

This process is illustrated in Figure 24. Briefly, the numerator reflects a probability of observing the next event from the current time (projected on the event gap histogram using the elapsed time since the most recent event time) to the new time defined by the window size W. The denominator defines a normalizer that takes into account the fact that no event has been seen during the time period defined by the elapsed time and basically corresponds to the probability of observing the event from the current time till the infinity. We assume that $r_t^e = 0$ if there is no prior occurrence of the event e.

4.2.3 Combining Predictive Signals

To summarize and utilize different aspects of complex clinical event time series, we aggregate information from multiple modules. As shown in Figure 25, we combine information from the periodicity module with distant past information from the neural abstraction module and recent information from the recent context module introduced in Chapter 3.

More specifically, the prior memory-based periodic signal p_t^e (Equation (21)) and current patient-specific periodic signal r_t^e (Equation (22)) are combined with the hidden states vector h_t (Equation (18)) and the recent context vector b_u (Equation (19)) from Chapter 3. With these predictive signals ready, we compute final output of the model as follows: First, we

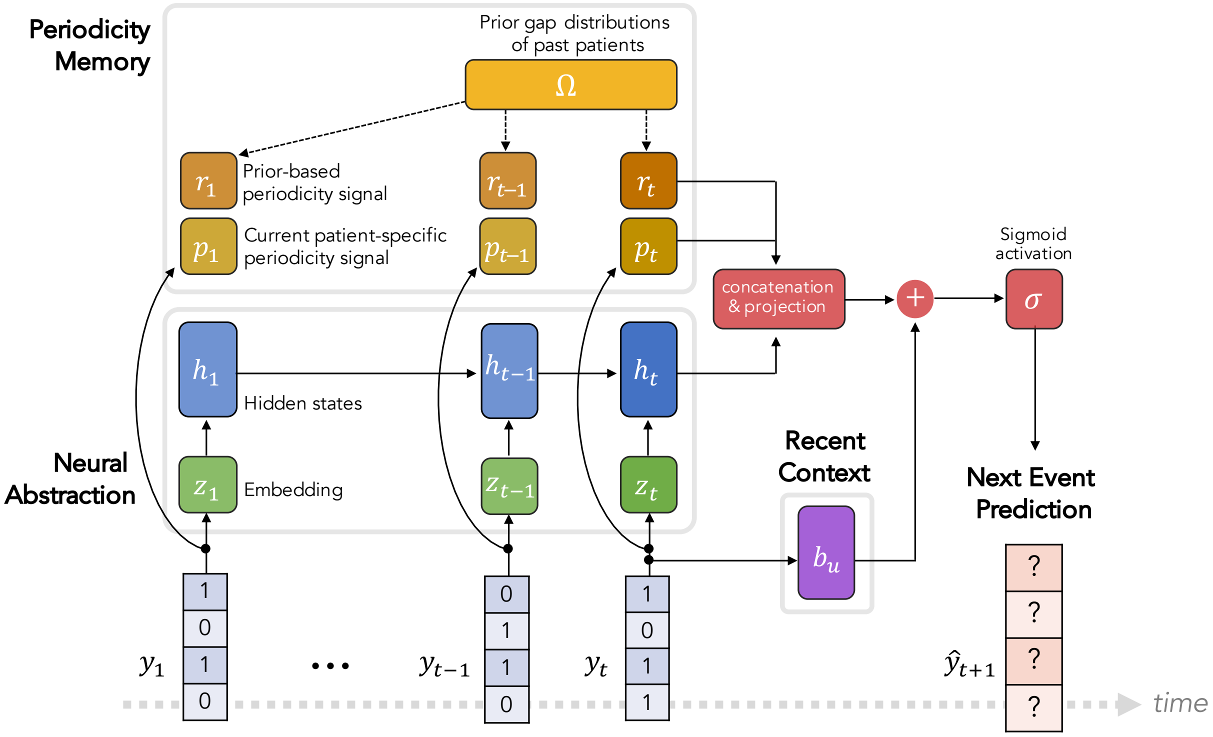


Figure 25: Architecture of the proposed model. Based on the architecture proposed in Chapter 3, information on recurring events are specifically modeled through the periodicity memory. With the periodicity memory module, we model different temporal aspects of information in the multivariate event time series $(y_1, \ldots, y_{t-1}, y_t)$ more comprehensively through the three different mechanisms: neural abstraction module for modeling long-term dependencies, recent context module for modeling recent dependencies and periodicity memory modules for modeling dependencies on previous recurrent event occurrences.

compute event-specific intermediate output $\tilde{o}^e$:

$$\tilde{o}^e = (W^e_{out} \cdot [h_t, p^e_t, r^e_t] + b^e_{out})$$

where $W^e_{out} \in \mathbb{R}^{1\times(h+2)}$ and $b^e_{out} \in \mathbb{R}$ are parameters of the linear transformation of the vector combining all signals. The final output for next event occurrence is computed as follows:

$$\hat{y}_{t+1} = \sigma([\tilde{o}^1, ..., \tilde{o}^{|E|}] + b_u) \tag{23}$$

The proposed predictor combines information on distant past from LSTM's hidden states and event gap-based information from periodicity module through concatenation and important signal for each event e is selected through linear regression parameterized with W^e_{out}, b^e_{out}. Then, the recent state (most recent events) information is added as an additional *recent bias* term to adjust information from LSTM's hidden states and the periodicity module with information from recent event occurrences.

4.2.4 Parameter Learning

As discussed in Chapter 3, the parameters W_{out}, b_{out} and the parameters of the neural abstraction and recent context modules are learned through an adaptive stochastic gradient descent based optimizer (Adam) [97] with binary cross entropy for the loss function. For the prior event gap distribution, the parameter is learned nonparametrically by counting and normalizing the histogram bins of each event-type.

4.3 Experimental Evaluation

In this section, we evaluate the performance of the proposed model on MIMIC III data [89] and compare it with baseline models introduced Chapter 3. Likewise, details of the experimental setup (including clinical data, feature preparation, baseline models, and evaluation metrics) are identical to the experiment setup introduced in Section 3.3 of Chapter 3.

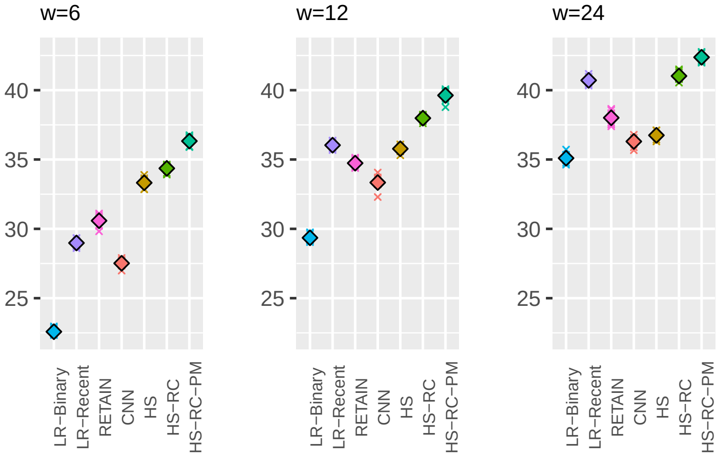


Figure 26: Overall prediction results. The results show average test AUPRCs over all events and 10 different random train-test splits.

4.3.1 Experiment Results

Figure 26 summarizes prediction results for all event types for three window sizes ($W = 6, 12, 24$) by averaging AUPRC obtained on our model and baselines. We can clearly see, that our model, HS-RC-PM, outperforms all baselines with a clear margin in all window segmentation settings.

Analyzing the results in Figure 26 by looking at the performance gap between HS-RC-PM and HS-RC, we can clearly see the added benefit of the periodicity module to the prediction performance since the two models differ exactly in the inclusion of that module. Digging deeper to understand this difference, Table 2 shows AUPRCs for some events in which HS-RC-PM brings remarkable enhancement in the predictive performance compared to HS-RC. Figure 27 shows the distribution of event gaps for these events (events in Table 2). Indeed, it is no surprise to observe such a performance improvement given strong periodicity in events. The events with clear periodic occurring behavior translate to the largest performance gap between HS-RC-PM and HS-RC.

4.3.1.1 Analysis of Results based on Repetition Patterns

The overall results showed the performance boost from the inclusion of the periodicity module. To further verify the effectiveness of the module across all events, we divide the event time series based on the number of previous events occurrences for each event type and compute the performance for each group. Briefly, we divide the event time series into three groups: (**G1**) time series with no previous event occurrences (from the beginning till the event occurs the first time), (**G2**) one previous event occurrence (after the event is observed first time till it is observed the second time), and (**G3**) two and more previous event occurrences (after observing the event second time and to the end of the time series). When properly trained, it is expected that the performance gap between HS-RC-PM and HS-RC should be visible for groups **G2** and mainly for **G3**, since the periodicity module is not able to generate any relevant signal until it gets the first event occurrence (case **G1**). Figure 28 shows the results for these three groups.

As expected, the performance gap between HS-RC-PM and HS-RC is widened at **G2** as we expected. This clearly reflects the value of the information on periodic events compiled through periodicity memory. Also, differences in the gap between HS-RC-PM and HS-RC in Figure 28b (**G2**) and Figure 28c (**G3**) shows how the patient-specific recent interval could be informative toward accurate prediction of the time series.

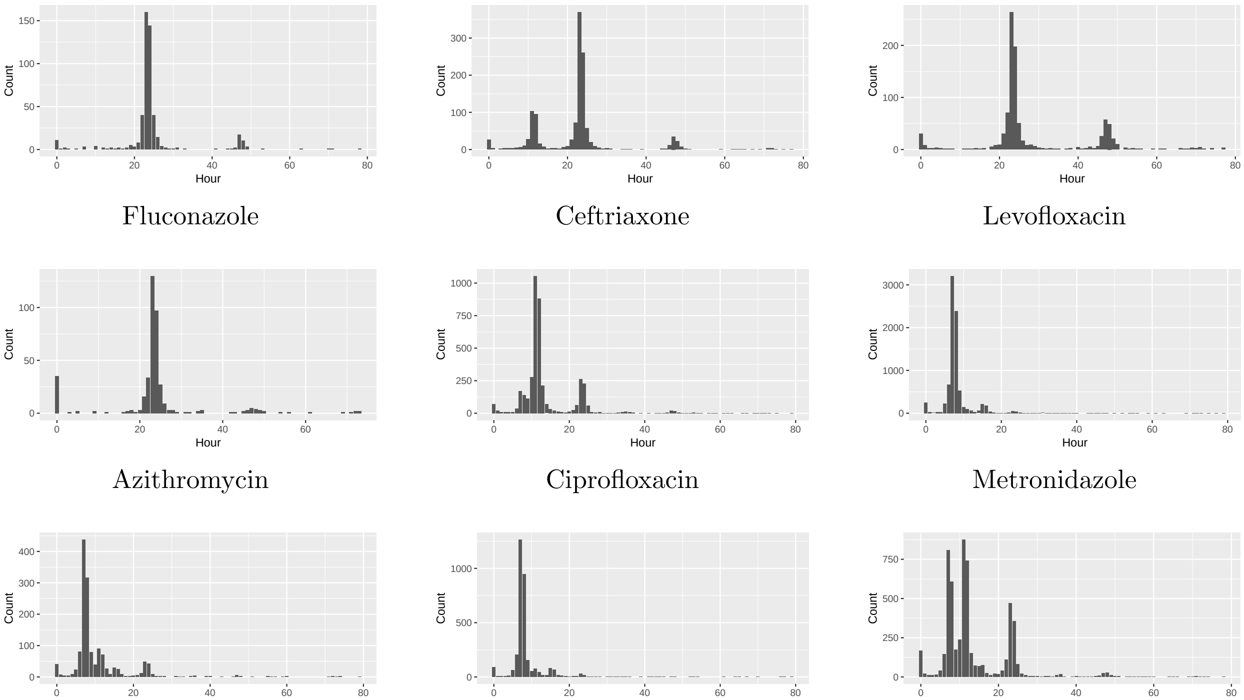
4.3.1.2 Analysis of Results based on Event Categories

To analyze the experiment results further, we next break the evaluation results down by inspecting predictive performances of the models for the four different event categories: medication events, lab events, physiological events, and procedure events. The results are shown in Figure 43. Clearly, HS-RC-PM consistently outperforms baseline models across all event categories in AUPRC statistics.

Notably, in the medication administration category, the performance gap between HS-RC-PM and other baselines is greater. It shows the periodicity module picks up the important signal on periodically occurring events. In ICU, medications often follow periodic or quasi-periodic administration regimes.

Event	**HS**	**HS-RC**	**HS-RC-PM**
[Med] Fluconazole	1.98	2.20	34.54
[Med] Ceftriaxone	5.40	4.79	33.02
[Med] Levofloxacin	3.27	3.27	26.80
[Med] Azithromycin	1.94	2.09	25.04
[Med] Ciprofloxacin	26.97	28.46	50.82
[Med] Metronidazole	52.78	49.43	70.86
[Med] Acyclovir	32.13	31.00	51.63
[Med] Cefazolin	43.69	41.50	60.74
[Med] Cefepime	36.55	35.82	54.38

Table 2: Performance on top 9 events with largest gap between HS-RC and HS-RC-PM (W=6)



Fluconazole

Ceftriaxone

Levofloxacin

Azithromycin

Ciprofloxacin

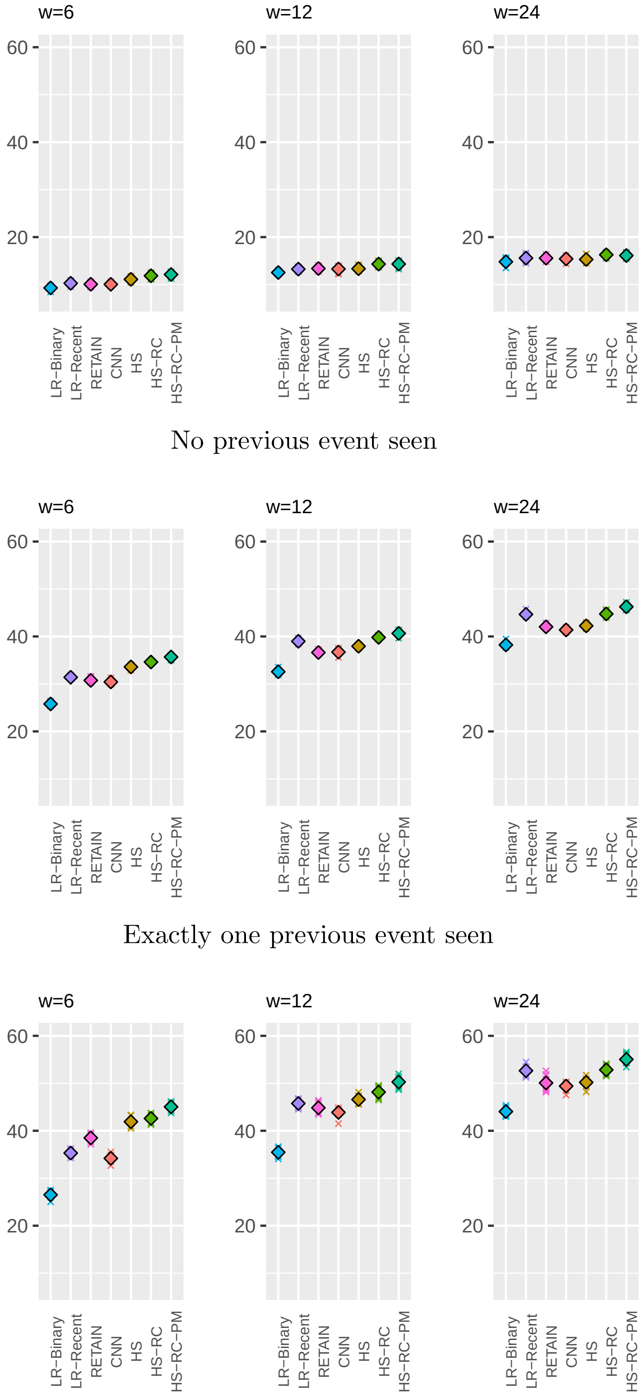
Metronidazole

Acyclovir

Cefazolin

Cefepime

Figure 27: Histograms of inter-event gaps of two consecutive occurrences of the top performing events shown in Table 2

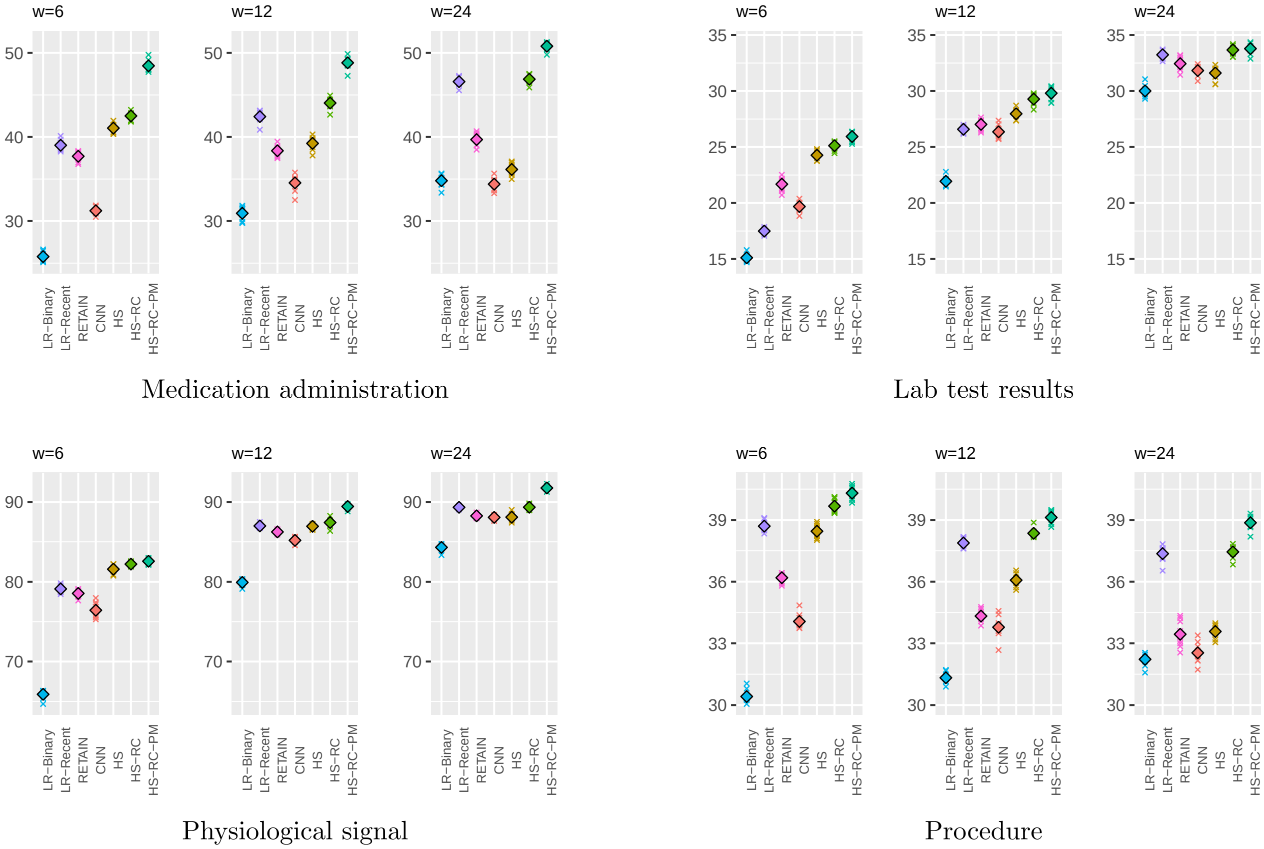


No previous event seen

Exactly one previous event seen

Two and more previous events seen

Figure 28: Prediction results based on the number of previous events seen



Medication administration

Lab test results

Physiological signal

Procedure

Figure 29: Prediction results by the event type category

4.4 Summary

In this chapter, we showed the importance of modeling periodic (repeated) events for predicting multivariate clinical event time series, in addition to modeling long term and recent events through the two modules introduced in Chapter 3. More specifically, we model periodic (repeated) events using a special external memory mechanism based on probability distributions of inter-event gaps compiled from past data. By combining predictive signals from the three modules, we enhance the modeling of multivariate event time series with different temporal mechanisms that aim to process different temporal aspects of the time series. We show that our model equipped with all the above temporal mechanisms leads to improved prediction performance compared to multiple baselines.

5.0 Modeling Clinical Event Time Series with Multi-scale Temporal Memory

5.1 Introduction

One challenging issue related to predictive EHRs representations that have not been adequately addressed is how to properly model temporal dependencies among many different clinical events. More specifically, individual event time series in EHRs may have a different temporal dependency with respect to precursor events. Briefly, some events may strongly dependent on recently occurred events. For example, an administration of phenylephrine depends on the occurrence of hypotension (low blood pressure state) in connection with recent intubation. Lee and Hauskrecht [108] show that modeling such short-term dependency can improve the predictability of multivariate future events. However, other events may depend on more distant events. For example, valve replacement surgery in the distant past may impact the necessity of warfarin treatment. While neural temporal models (RNN or LSTM) can in principle model these long-range dependencies, the recurrent computations can easily dilute and attenuate such information in the hidden state [167]. In this chapter, we address the problem of modeling long-term dependencies in multivariate clinical event time-series by proposing a new type of information channel linking events in a distant past with the current prediction time. Through a novel mechanism called Multi-scale Temporal Memory (MTM), information about previous events on different time-scales is compiled and read on-the-fly for prediction through memory contents. The main benefit of this approach is that it is a modular and predictive signal from this module that can be combined with predictive signals from other patient state summarization modules.

We demonstrate the efficacy of MTM by combining it with different patient state summarization methods that cover different temporal aspects of patient states, including recent context module Chapter 3, recurrent temporal mechanism Chapter 4, and hidden states of LSTM [77]. We test the proposed approach on real-world clinical event time-series. We compare predictive performance (i.e., AUPRC) of the proposed combined approach with baseline models. We demonstrate that the combined approach is 4.6% more accurate than

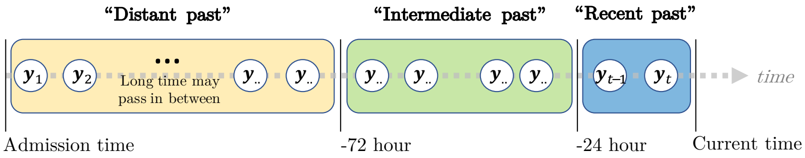


Figure 30: MTM summarizes past history with multiple temporal scales

the best among the baselines and it is 16% more accurate than prediction solely through hidden states of LSTM.

5.2 Methodology

We propose Multi-scale Temporal Memory (MTM), a new neural temporal based model that summarizes a clinical event history and generates a predictive signal for occurrence and non-occurrence of future multivariate clinical events.

5.2.1 Multi-Scale Temporal Memory

MTM summarizes patient history using multiple information channels where each channel covers the history in different temporal scales. We hypothesize that information on past event occurrences on different time range may have different importance for predicting future event occurrence. To process patient history on multiple time scales, MTM segments the patient history into three folds as shown in Figure 30: distant past (e.g., from the beginning of admission to 72 hours before current time), recent past (e.g., within 24 hours from current time), and ”intermediate past” (time range between boundaries of distant past and recent past). Contents of the memory are composed based on the types of events that occurred in each segmented window. Then, considering factors about current patient states, the model reads contents of the multi-scale memory and generates a predictive signal that will be combined with other neural temporal mechanisms that cover different aspects of clinical event time-series to generate a final prediction for next multivariate events. In the following, we describe MTM in detail and the neural framework for the next multivariate events prediction.

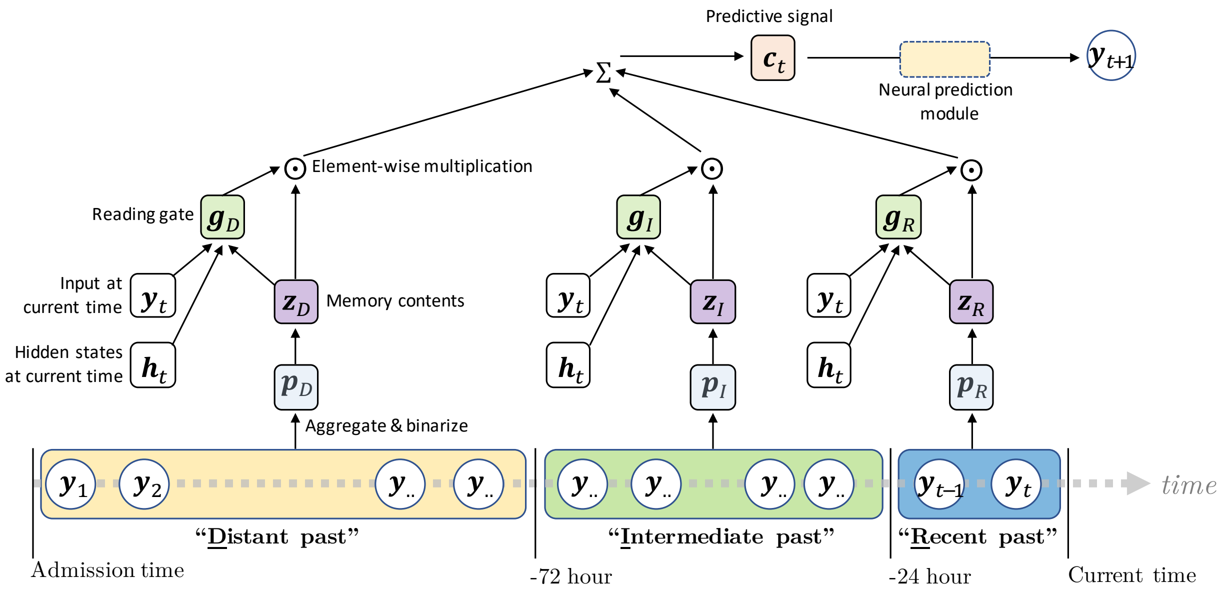


Figure 31: Overview MTM's architecture: Given a sequence of multivariate patient state history $\mathbf{y}_1, \ldots, \mathbf{y}_t$, we **(1)** aggregate and binarize past history by each time-scale $\mathbf{p}_*, * \in \{D, I, R\}$, **(2)** compose memory contents $\mathbf{z}_*$, **(3)** compute reading gate $\mathbf{g}_*$, **(4)** read memory contents referring reading gate and merge contents of multi-scale temporal memory, and **(5)** make a predictive signal $\mathbf{c}_t$ for neural prediction module.

5.2.1.1 Composing Memory Contents

Given a segmented patient history (depicted in Figure 30) on multiple time-scales, we compose memory contents for each time-scale with the following steps: (1) We aggregate patient states vectors $\{\mathbf{y}_i\}_i$ of each temporal segment $* \in \{D, I, R\}$ into a single multivariate vector $\mathbf{p}_*$ through binarization. $\{D, I, R\}$ denote distant, intermediate, and recent pasts respectively. (2) We compose contents of the memory $\mathbf{z}_* \in \mathbb{R}^{|E|}$ through linear projection followed by non-linear activation:

$$\mathbf{z}_* = \tanh\left(\mathbf{W}_*\mathbf{p}_* + \mathbf{b}_*\right) \tag{24}$$

where $\mathbf{W}_* \in \mathbb{R}^{|E|\times|E|}$ and $\mathbf{b}_* \in \mathbb{R}^{|E|}$ are trainable parameters for each time-scale. Through linear projection with $\mathbf{W}_*$, we extract information about the events that occurred in a specific temporal segment.

5.2.1.2 Reading Memory Contents

To comprehensively determine the amount of memory contents to be read for each prediction task (multivariate target events), MTM computes reading gates $\mathbf{g}_* \in \mathbb{R}^{|E|}$ considering three factors: (1) current patient state reflected on input $\mathbf{y}_t$, (2) recent dynamics of patient state reflected on hidden states $\mathbf{h}_t$ from LSTM, and the contents of the memory itself $\mathbf{z}_*$.

$$\mathbf{g}_* = \sigma(\mathbf{W}_h\mathbf{h}_t + \mathbf{W}_y\mathbf{y}_t + \tilde{\mathbf{W}}_*\mathbf{z}_*) \tag{25}$$

where σ denotes logistic sigmoid activation function and $\mathbf{W}_h \in \mathbb{R}^{|E|\times r}, \mathbf{W}_y \in \mathbb{R}^{|E|\times|E|}, \tilde{\mathbf{W}}_* \in \mathbb{R}^{|E|\times r}$ are parameters to learn and r is dimension of hidden state. The predictive signal $\mathbf{c}_t \in \mathbb{R}^{|E|}$ is computed as a linear combination of reading gates and memory contents for each temporal scale:

$$\mathbf{c}_t = \mathbf{g}_D \odot \mathbf{z}_D + \mathbf{g}_I \odot \mathbf{z}_I + \mathbf{g}_R \odot \mathbf{z}_R \tag{26}$$

where $\odot$ is element-wise multiplication.

5.2.2 Neural-based Prediction Framework

We combine the predictive signal from MTM with additional patient history summarization methods that cover different temporal aspects of patient states. We use recent-context module [108] in Chapter 3, recurrent temporal mechanism [109] in Chapter 4 and hidden states of LSTM. Briefly the recent-context module projects current time-step input $\mathbf{y}_t$ to a target event space with a learnable parameters $\mathbf{W}_r \in \mathbb{R}^{|E|\times|E|}$ and $\mathbf{b}_r$ to get the "recent bias" term $\mathbf{b}_\kappa$:

$$\mathbf{b}_\kappa = \mathbf{W}_r \mathbf{y}_t + \mathbf{b}_r \tag{27}$$

The recurrent temporal mechanism captures information about periodic (repeated) events using a special recurrent mechanism based on probability distributions of inter-event gaps. It outputs two target event-specific periodicity-based predictive signals that use different sources of periodic information: $\boldsymbol{\alpha}^e \in \mathbb{R}$ signal is based on an interval of current patient's event time-series and $\boldsymbol{\beta}^e \in \mathbb{R}$ signal is compiled from a pool of past patient data in training set. Details of the signal generation processes can be found in [109]. We also use LSTM to derive dynamics of patient state through hidden state. To compute hidden state, we first project input $\mathbf{y}_t$ to low-dimensional space with embedding matrix: $\mathbf{W}_{emb} \in \mathbb{R}^{d \times |E|}$: $\mathbf{x}_t = \mathbf{W}_{emb}\mathbf{y}_t$. Based on previous time step's hidden state $\mathbf{h}_{t-1}$ and $\mathbf{x}_t$, we compute new hidden state $\mathbf{h}_t \in \mathbb{R}^r$:

$$\mathbf{h}_t = \mathrm{LSTM}(\mathbf{h}_{t-1}, \mathbf{x}_t) \tag{28}$$

Given predictive signals $\{\boldsymbol{\alpha}^e, \boldsymbol{\beta}^e, \mathbf{h}_t, \mathbf{c}_t, \mathbf{b}_\kappa\}$, we first combine periodicity-based signals for each target event type with hidden state through concatenation:

$$\boldsymbol{\gamma}^e = [\mathbf{h}_t; \boldsymbol{\alpha}^e; \boldsymbol{\beta}^e] \tag{29}$$

Then, we project $\boldsymbol{\gamma}^e$ to a scalar $\lambda^e \in \mathbb{R}$ through $\mathbf{w}_e \in \mathbb{R}^{1 \times r+2}$ and $b_e \in \mathbb{R}$. We apply the same procedure to all events $e \in E$ and concatenate all λ^e:

$$\lambda^e = \mathbf{w}_e \boldsymbol{\gamma}^e + b_e \quad \boldsymbol{\lambda} = [\lambda^1; \dots ; \lambda^{|E|}] \tag{30}$$

Final prediction for next multivariate event is computed as follows:

$$\hat{\mathbf{y}}_{t+1} = \sigma(\boldsymbol{\lambda} + \mathbf{b}_\kappa + \mathbf{c}_t) \tag{31}$$

We use the binary cross-entropy to compute loss $\mathcal{L}$ and parameters of the model are learned through a stochastic gradient descent optimization algorithm (Adam) [97].

$$\mathcal{L} = \sum_t -[\mathbf{y}_t \cdot \log \hat{\mathbf{y}}_t + (\mathbf{1} - \mathbf{y}_t) \cdot \log(\mathbf{1} - \hat{\mathbf{y}}_t)] \tag{32}$$

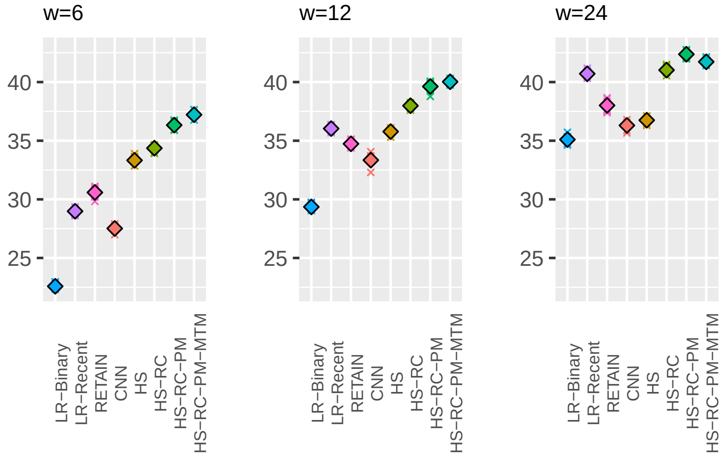


Figure 32: Overall prediction results. The results show average test AUPRCs over all events and 10 different random train-test splits.

5.3 Experimental Evaluation

In this section, we evaluate the performance of the proposed model on MIMIC III data [89] and compare it with baseline models introduced Chapter 3 and Section 4.2. Details of the experimental setup (including clinical data, feature preparation, baseline models, and evaluation metrics) are identical to the experiment setup introduced in Section 3.3 of Chapter 3.

5.3.1 Experiment Results

Figure 32 shows the overall experiment results for predicting all types of events for three window sizes (W=6,12,24). The proposed model (HS-RC-PP-MTM) outperforms all baselines for shorter window sizes (W=6,12). Particularly, it outperforms HS-RC-PP by 2.4+% in W=6. With this, we can observe the benefit of multi-scale memory capturing dependencies that are not covered by other patient history summarization methods, including LSTM in much longer sequence setting.

We further analyze the experiment results by dividing them into 4 event categories. As

Models	W=1	W=6	W=12
HS-RC-PP	26.76	36.68	40.07
HS-RC-PP-MTM	28.00 (4.6 +%)	37.28 (1.6 +%)	40.34 (0.6 +%)

Table 3: Prediction results (AUPRC) by varying time-series segmentation window settings. In a shorter segmentation window ($W = 6$) we see higher performance gain by the proposed MTM module.

shown in Figure 33, we observe the performance gain of MTM is higher for medication and lab test events. Notably, lab tests are the hardest events to predict compared to other categories, 14+% performance gain from MTM for lab test prediction clearly shows its effectiveness.

To validate learned weight matrices for multi-scale memory contents ($W_*, * \in \{D, I, R\}$ in Equation (24)), we extract the top 3 events for exemplar target events

$$\text{argsort}(W_*[i, :])[1 : 3]$$

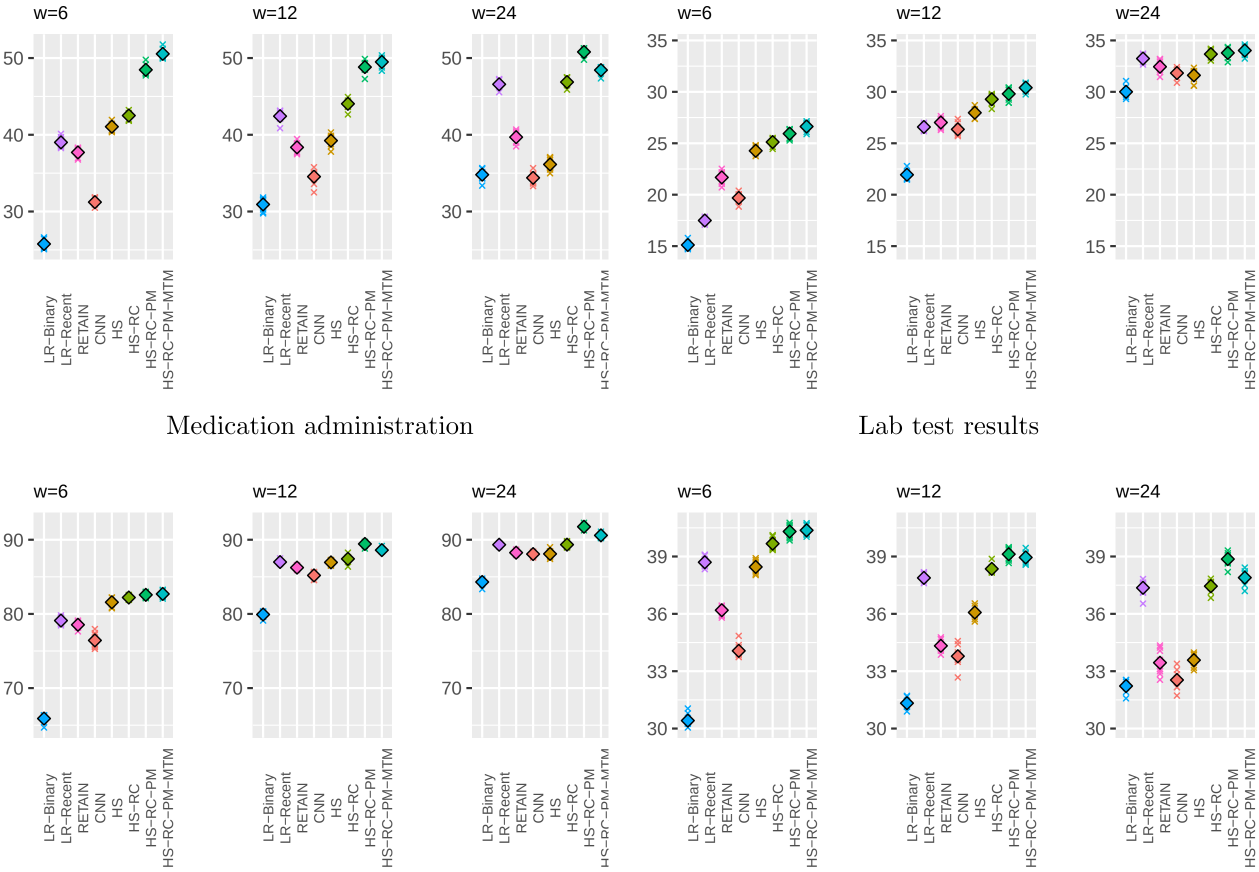
where i represents index for a target event in the matrix. As shown in Table 4, the top predictive events for amiodarone (treats irregular heartbeat such as tachycardia) include metoprolol and diltiazem. Both of these are used to treat high blood pressure and heart issues. Similarly, past events predictive of diltiazem and labetalol (medications treating high blood pressure) include clinical events that are related to high blood pressure and heart function: digoxin, metoprolol, hydralazine, and nicardipine. Finally, the top past events predicting vasopressin (a medication treating a low blood pressure) include norepinephrine and phenylephrine that are also used to treat low blood pressure.

5.4 Summary

In this chapter, we proposed a novel mechanism called Multi-scale Temporal Memory (MTM) to model long-term dependencies in EHR-derived clinical event time-series. With MTM, information about past events on different time-scales is compiled and read on-the-fly for prediction through memory contents. We demonstrate the efficacy of MTM by combining it with different patient state summarization methods that cover different temporal aspects of patient states. We show that the combined approach is 4.6% more accurate than the baseline approaches and it is 16% more accurate than the prediction based on the popular LSTM summarization approach.

Distant past ($*$=D)	**Intermediate past**($*$=I)	**Recent past**($*$=R)
Target: (Med) Amiodarone		
(Med) Amiodarone	(Med) Amiodarone	(Med) Amiodarone
(Med) Diltiazem	(Med) Diltiazem	(Med) Metoprolol
(Lab) Urea Nitrogen, Urine	(Lab) Thyroid Stimulating Hormone	(Med) Diltiazem
Target: (Med) Diltiazem		
(Med) Diltiazem	(Med) Diltiazem	(Med) Diltiazem
(Lab) Digoxin	(Med) Metoprolol	(Med) Metoprolol
(Physio) Inspired O2 Fraction	(Med) Amiodarone	(Proc) EKG
Target: (Med) Labetalol		
(Med) Labetalol	(Med) Labetalol	(Med) Labetalol
(Med) Hydralazine	(Med) Hydralazine	(Med) Hydralazine
(Med) Nicardipine	(Med) Metoprolol	(Med) Haloperidol
Target: (Med) Vasopressin		
(Med) Vasopressin	(Med) Vasopressin	(Med) Vasopressin
(Proc) Ultrasound	(Med) Norepinephrine	(Med) Norepinephrine
(Med) Packed Red Blood Cells	(Med) Phenylephrine	(Med) Phenylephrine

Table 4: Top 3 preceding events for example target events based on the value from learned memory content parameter W_* for each temporal range in Equation (24).



Medication administration

Lab test results

Physiological signal

Procedure

Figure 33: Prediction results by the event type category

6.0 Learning to Adapt Dynamic Clinical Event Sequences with Residual Mixture of Experts

6.1 Introduction

As discussed in the Section 1.4, one important challenge of learning highly accurate models of clinical sequences is patient-specific variability. Depending on the underlying clinical condition specific to a patient combined with multiple different management options one can choose and apply in patient care, the event patterns may vary widely from patient to patient. Unfortunately, many modern event prediction models and assumptions incorporated into the training of such models may prevent one from accurately representing such a variability. In Chapter 6 and Chapter 7, we want to address this challenge by developing two different approaches that adapt the event predictions to represent better individual patient-specific behaviors and event sequences.

In this chapter, we study ways of enhancing the one-model solution to adapt to the heterogeneity of overall patient population and its subpopulations. We study this solution in context of multivariate event prediction problem where our goal is to predict as accurately as possible the occurrence of a wide range events recorded in EHRs. Such a prediction task can be used for defining general patient state representation that can be used for example to define similarity among patients or for predicting the patient outcomes. To adapt to different subpopulations and their behaviors we explore the mixture of experts architecture [83]. In other words, our model aims to learn one primary autoregressive model that is then adapted to different subpopulations using the mixture of experts architecture. The mixture attempts to represent many residual models refining the all-population model. The benefit of such a model is that the subpopulation models may be much simpler than the original population models. To make the mixture capable of refining the all-population model, we take a different approach to train the mixture of experts. Instead of directly training the mixture from scratch, we first train the all-population model, and while fixing the parameters of pretrained the all-population model, we train the proposed model, which combines the all population model's output and the mixture of experts output. With this way, the mixture can learn to adapt the residual of the all-population model. Hence we name our model Residual Mixture of Experts (R-MoE). R-MoE provides flexible adaptation to the (limited) predictive power of the all-population model.

We demonstrate the effectiveness of R-MoE on the task of multivariate clinical sequence prediction which uses real-world patient data from MIMIC-3 Database [89]. R-MoE shows 4.1% gain on AUPRC compared to a single GRU-based prediction.

6.2 Methodology

6.2.1 Neural Event Sequence Prediction

In this chapter, our goal is to predict the occurrences and non-occurrences of future clinical (target) events $\boldsymbol{y}'_{t+1}$ for a patient given the patient's past clinical event occurrences $\boldsymbol{H}_t$. Specifically, we assume that the patient's clinical event history is in a sequence of multivariate input event vectors $\boldsymbol{H}_t = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t\}$ where each vector $\boldsymbol{y}_i$ is a binary $\{0, 1\}$ vector, one dimension per an event type. The input vectors are of dimension $|E|$ where E are different event types in clinical sequences. The target vector is of dimension $|E'|$, where $E' \subset E$ are events we are interested in predicting. We aim to build a predictive model δ that can predict $\hat{\boldsymbol{y}}'_{t+1}$ at any time t given the history $\boldsymbol{H}_t$.

One way to build δ is to use neural sequence models such as RNN and LSTM. In this work, we use a bi-directional attention mechanism model (RETAIN) [38] to build a base prediction model δ_{base}. RETAIN is a representative work on using attention mechanism to summarize clinical event sequences, proposed by Choi et al. [38]. It uses two attention mechanisms to comprehend the history of GRU-based hidden states in reverse-time order. Besides parameters RETAIN, we also have input embedding matrix $\boldsymbol{W}_{emb} \in \mathbb{R}^{|E| \times \epsilon}$, output projection matrix $\boldsymbol{W}_{out} \in \mathbb{R}^{d \times |E'|}$, bias vector $\boldsymbol{b}_{out} \in \mathbb{R}^{|E'|}$, and a sigmoid (logit) activation function σ. At any time step t, we update update hidden state $\boldsymbol{h}_t$, and predict target events in next time step $\hat{\boldsymbol{y}}'_{t+1}$:

$$\begin{aligned}\boldsymbol{v}_t &= \boldsymbol{W}_{emb} \cdot \boldsymbol{y}_t \\ \boldsymbol{h}_t &= \text{RETAIN}(\boldsymbol{h}_{t-1}, \boldsymbol{v}_t) \\ \hat{\boldsymbol{y}}'_{t+1} &= \sigma(\boldsymbol{W}_o \cdot \boldsymbol{h}_t + \boldsymbol{b}_o)\end{aligned} \tag{33}$$

The all parameters of δ_{base} ($\boldsymbol{W}_{emb}, \boldsymbol{W}_o, \boldsymbol{b}_o$ and RETAIN) are learned through stochastic gradient descent (SGD) algorithm with binary cross entropy loss function.

This RETAIN-based neural sequence model has a number of benefits for modeling complex high-dimensional clinical event time series: First, we can obtain a compact real-valued representation of high-dimensional binary input vector $\boldsymbol{y}$ through low-dimensional embedding with $\boldsymbol{W}_{emb}$. Second, we can model complex dynamics of patient state sequences through RETAIN which can model non-linearities of the sequences with attention mechanism. Third, complex input-output associations of the patient state sequences can be learned through a flexible SGD-based end-to-end learning framework.

Nonetheless, the neural approach cannot address one important peculiarity of the patient state sequence: the heterogeneity of patient sequences. Typically, clinical event sequences in EHRs are generated from a pool of diverse patients where each patient have different types of clinical complications, medication regime, or observed sequence dynamics. While the average behavior of clinical event sequences can be captured by a single neural sequence model, the detailed dynamics of heterogeneous clinical event sequences could not be well captured.

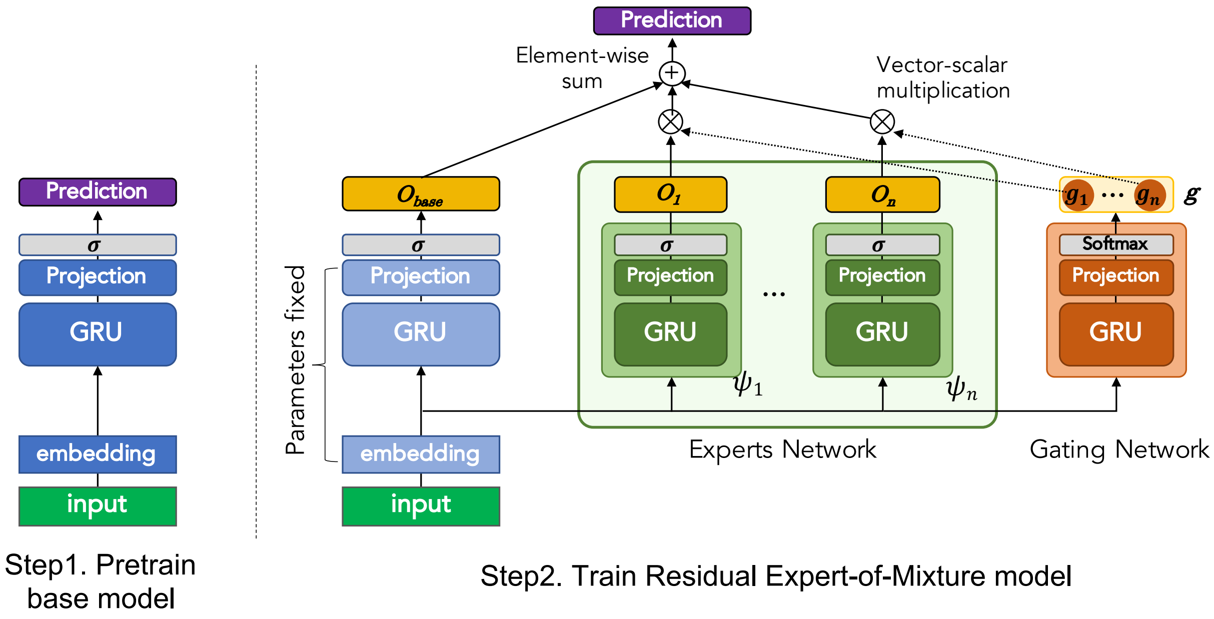


Figure 34: Overall architecture of the proposed R-MoE model. First we train base model λ_{base}. Then, we fix the parameters of λ_{base} and train the parameters of the Mixture-of-Experts consists of Experts network and Gating network with the combined prediction of λ_{base} and the MoE. With this way, MoE can learn to adapt the residual of λ_{base}.

6.2.2 Residual Mixture-of-Experts

In this work, we address the heterogeneity issue of the neural sequence model by specializing it with a novel learning mechanism based on Mixture-of-Experts (MoE) architecture. The dynamics of heterogeneous patient state sequences can be modeled through a number of experts; each consists of GRU, which is capable of modeling non-linearities and temporal dependencies. Particularly, in this work, instead of simply replacing the GRU model δ_{base} with MoE, we *augment* δ_{base} with MoE. The key idea is to specialize the Mixture-of-Experts to learn the residual which δ_{base} cannot capture. As shown in Figure 34, the proposed model R-MoE consists of δ_{base} module and Mixture-of-Experts module.

The Mixture-of-Experts module consists of n experts $\psi_1, \ldots, \psi_n$ and a gating network

G which outputs a n-dimensional vector $\boldsymbol{g}$. The output $\boldsymbol{o}_{moe}$ of the MoE module can be written as follows:

$$\boldsymbol{o}_{moe} = \sum_{i=1}^{n} \boldsymbol{g}_{[i]}(\boldsymbol{v}_t) \cdot \psi_i(\boldsymbol{v}_t) \tag{34}$$

Each expert ψ_i consists of GRU, output projection matrix $\boldsymbol{W}_o^i \in \mathbb{R}^{d' \times |E'|}$, and a bias vector $\boldsymbol{b}_o^i \in \mathbb{R}^{|E'|}$. Given an input in low-dimensional representation $\boldsymbol{v}_t$, an expert ψ_i outputs $\boldsymbol{o}^i$:

$$\boldsymbol{h}_t^i = \mathrm{GRU}^i(\boldsymbol{h}_{t-1}^i, \boldsymbol{v}_t) \qquad \boldsymbol{o}^i = \sigma(\boldsymbol{W}_o^i \cdot \boldsymbol{h}_t^i + \boldsymbol{b}_o^i) \qquad i \in 1, \ldots, n$$

The gating network G have the same input and a similar architecture, except that its output $\boldsymbol{g}$'s dimension is n and it is through $Softmax$ function. $\boldsymbol{g}_{[i]}$ in Equation (34) represents i value in the vector $\boldsymbol{g}$.

$$\boldsymbol{h}_t^g = \mathrm{GRU}^g(\boldsymbol{h}_{t-1}^g, \boldsymbol{v}_t) \qquad \boldsymbol{g} = Softmax(\boldsymbol{W}_o^g \cdot \boldsymbol{h}_t^g + \boldsymbol{b}_o^g)$$

The final prediction $\hat{\boldsymbol{y}}'_{t+1}$ is generated by summing outputs of the two modules $\boldsymbol{o}_{base} = \delta_{base}(\boldsymbol{H}_t)$ and $\boldsymbol{o}_{moe}$:

$$\hat{\boldsymbol{y}}'_{t+1} = \boldsymbol{o}_{base} + \boldsymbol{o}_{moe} \tag{35}$$

To properly specialize the Mixture-of-Experts on the residual, we train the two modules as follows: First, we train δ_{base} module, and parameters of δ_{base} are fixed after the train. Then, we train the MoE module with the binary cross entropy loss computed with the final prediction in Equation (35). With this way, MoE can learn to adapt the residual, which the base GRU cannot properly model. MoE provides flexible adaptation to the (limited) predictive power of the base GRU model.

6.3 Experimental Evaluation

6.3.1 Experiment Setup

In this section, we evaluate the performance of R-MoE model on the real-world EHRs data in compare it with four baseline models. Likewise, most details of the experimental setup (including clinical data, feature preparation, and evaluation metrics) are identical to the experiment setup introduced in Section 3.3 of Chapter 3. In this chapter, we focus on three baseline models which are presented below with details of model parameters.

6.3.1.1 Baseline Models

We compare R-MoE with multiple baseline models that are able to predict events for multivariate clinical event time series given their previous history. The baselines are:

- **Base GRU model (GRU)**: GRU-based event time series modeling described in Equation (33). (λ=1e-05)
- **REverse-Time AttenTioN (RETAIN)**: RETAIN is a representative work on using attention mechanism to summarize clinical event sequences, proposed by Choi et al. [38]. It uses two attention mechanisms to comprehend the history of GRU-based hidden states in reverse-time order. For multi-label output, we use a sigmoid function at the output layer. (λ=1e-05)
- **Logistic regression based on Convolutional Neural Network (CNN)**: This model uses CNN to build predictive features summarizing the event history of patients. Following Nguyen et al. [159], we implement this CNN-based model with a 1-dimensional convolution kernel followed by ReLU activation and max-pooling operation. To give more flexibility to the convolution operation, we use multiple kernels with different sizes (2,4,8) and features from these kernels are merged at a fully-connected (FC) layer. (λ=1e-05)

6.3.1.2 Model Parameters

We use embedding dimension ϵ=64, hidden state dimension d=512 for base GRU model and RETAIN. Hidden states dimension d' for each GRU in R-MoE is determined by the internal cross-validation set (range: 32, 64, 128, 256, 512). The number of experts for R-MoE is also determined by internal cross-validation set (range:1, 5, 10, 20, 50, 100). For the SGD optimizer, we use Adam [97]. Learning rate for GRU, RETAIN, and CNN we use 0.005 and for R-MoE we use 0.0005. To prevent over-fitting, we use L2 weight decay regularization during the training of all models and weight λ is determined by the internal cross-validation set. Range of λ for GRU, RETAIN, and CNN is set as (1e-04, 1e-05, 1e-06, 1e-07). For R-MoE, after observing it requires much larger λ, we set the range of λ for R-MoE as (0.75, 1.0, 1.25, 1.5). We also use the early stopping to prevent over-fitting. That is, we stop the training when internal validation set's evaluation metric does not improve during last K epochs (K=5).

6.3.2 Results

Table 5 summarizes the performance of R-MoE and baseline models. The results show that R-MoE clearly outperforms all baseline models. More specifically, compared to RETAIN, the best-performing baseline model, our model shows 2.97% improvement. Compared to averaged AUPRC of all baseline models, our model shows 7.84% gain in AUPRC.

	CNN	RETAIN	GRU	R-MoE
AUPRC	35.71 ±0.12	38.64 ±0.22	36.34 ±0.14	39.79 ±0.24

Table 5: Prediction results of all models averaged over all events and 10 different random train-test splits.

To more understand the effectiveness of R-MoE, we look into the performance gain of our model at the individual event type level. Especially we analyze the performance gain along with the individual event type's occurrence ratio, which is computed based on how many times each type of event occurred among all possible segmented time-windows across all test set patient admissions. As shown in Figure 35, we observe more performance gains are among the events that less occurred.

6.3.2.1 Model Capacity and Performance of R-MoE

To further understand the performance of R-MoE regarding the various model capacities in terms of different numbers of experts and different dimensions of hidden states. Note that as written in Section 4.1 the best hyperparameter is searched through internal cross-

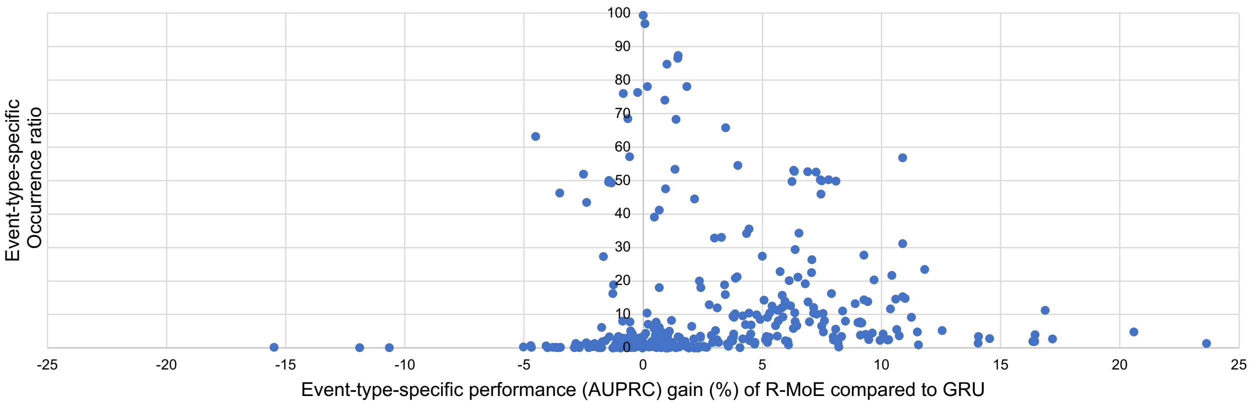


Figure 35: Event-type-specific AUPRC performance gain of R-MoE compared to the base RETAIN model (λ_{base}) and event-specific occurrence ratio. Each point represents each target event type among E. Occurrence ratio is how much times each event occurred among all segmented time-windows across all test set patient admissions.

validation (number of experts = 50 and hidden states dimension d'=64). Then, for this analysis, we fix one parameter at its best and show how the performance of R-MoE in another parameter by varying model capacity. Regarding the number of experts, a critical performance boost has occurred with a very small number of experts. As shown in Figure 36, with simply five experts, we observe 2.63% AUPRC gain compared to the baseline RETAIN model. With more experts, the performance is increasing, but the increment is very small after 50 experts. Regarding different hidden states dimensions of GRU (d') in R-MoE, we observe changing it does not affect much of the difference in predictive performance as shown in Figure 37.

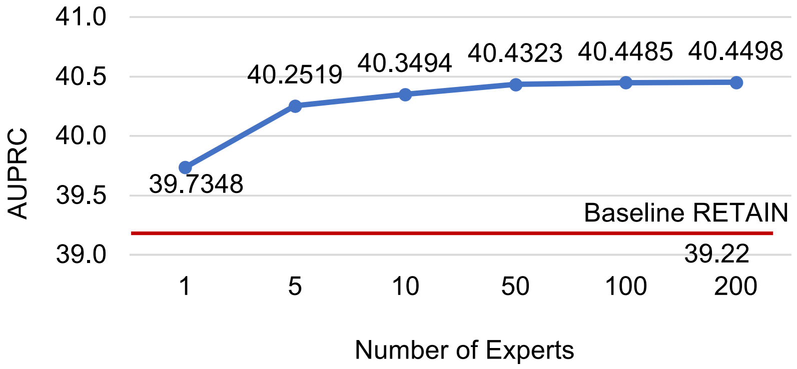


Figure 36: Prediction performance of R-MoE on different number of experts. Dimension of hidden states is fixed at 64.

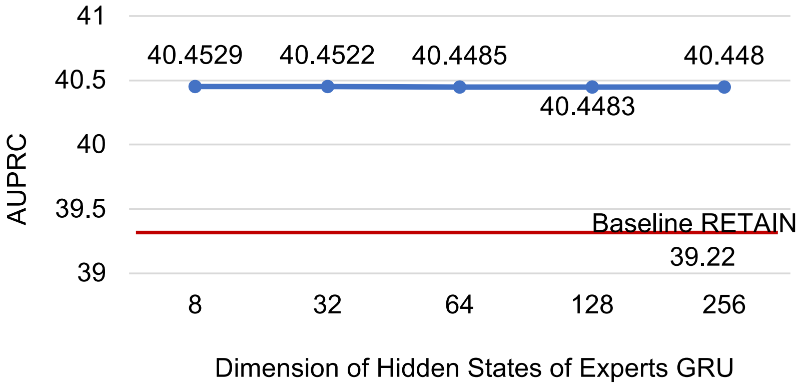


Figure 37: Prediction performance of R-MoE on different hidden states dimensions. Number of experts is fixed at 50.

6.4 Summary

In this chapter, we have developed a novel learning method that can enhance the performance of predictive models of multivariate clinical event sequences, which are generated from a pool of heterogeneous patients. We address the heterogeneity issue by introducing the Residual Mixture-of-Experts model. We demonstrate the enhanced performance of the proposed model through experiments on electronic health records for intensive care unit patients.

7.0 Modeling Clinical Event Sequences with Personalized Online Adaptive Learning Framework

7.1 Introduction

For the goal of learning personalized patient dynamic representation that can address the variability of the heterogeneous patient event sequences, we studied multiple sequential experts models that learn to adjust the population model's prediction in Chapter 6.

This chapter aims to develop a more straightforward approach that adjusts the population model's prediction through patient-specific prediction models trained on each patient's own past event history and similar other patients' histories available in the train set.

When neural event prediction models are built from complex multivariate clinical event sequences, the neural models may fail to accurately model patient-specific variability due to their limited ability to represent distributions of dynamic event trajectories. Briefly, the parameters of neural temporal models are learned from many patients data through Stochastic Gradient Descent (SGD) and are shared across all types of patient sequences. Hence, the population-based models tend to average out patient-specific patterns and trajectories in the training sequences. Consequently, they are unable to predict all aspects of patient-specific dynamics of event sequences and their patterns accurately.

To address the above problem, we propose, develop, and study two novel event time series prediction solutions that better adapt the population models to the individual patient. First, we propose a model that aims to improve a prediction made for the current patient at any specific time using a repository of event sequences recorded for past patients. The model works by first identifying the patient states among past patients that are most similar to the current state of the current patient and then adapting the predictions of the population-wide model with the help of outcomes recorded for such patients and their states. We refer to this model as the *subpopulation model.* Second, we develop and study a model that adapts the predictions of the population-wide model only based on the patients' own sequence. We refer to this model as the *self-adaptation model.* However, one concern with either the sub-population or the self-adaptation model and related adaptation is that it may lose some flexibility by being fit too tightly to the specific patient (and patient's recent condition) or to the patient state most similar to the current state. To address this, we also develop and investigate the meta-switching framework that is able to dynamically identify and switch to the best model to follow for the current patient. Briefly, the meta-framework uses a set of models and learns how to switch to the model offering the most promising solution adaptively. Such a framework may combine the population, subpopulation, and self-adaptation models. We note that all of the above solutions can extend RNN-based multivariate sequence prediction to support personalized clinical event sequence adaptation. We demonstrate the effectiveness of both solutions on clinical event sequences derived from real-world EHRs data from MIMIC-3 Database [89].

7.2 Methodology

7.2.1 Neural Autoregressive Event Sequence Prediction

Our goal is to predict occurrences of multiple target events in clinical event sequences. We aim to build an autoregressive model ϕ that can predict, at any time t, the next step (target) event vector $\boldsymbol{y}'_{t+1}$ from a history of past (input) event vectors $\boldsymbol{H}_t = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t\}$, that is, $\hat{\boldsymbol{y}}'_{t+1} = \phi(\boldsymbol{H}_t)$. The event vectors are binary $\{0, 1\}$ vectors, one dimension per an event type. The input vectors are of dimension $|E|$ where E are different event types in clinical sequences. The target vector is of dimension $|E'|$, where $E' \subset E$ are events we are interested in predicting.

One way to build a neural autoregressive prediction model ϕ is to use Recurrent Neural Network (RNN) with input embedding matrix $\boldsymbol{W}_{emb}$, output linear projection matrix $\boldsymbol{W}_o$, bias vector $\boldsymbol{b}_o$, and sigmoid (logit) activation function σ. At each time step t, the RNN-based autoregressive model ϕ reads new input $\boldsymbol{y}_t$, updates hidden state $\boldsymbol{h}_t$, and generates prediction of the target vector $\hat{\boldsymbol{y}}'_{t+1}$:

$$\boldsymbol{v}_t = \boldsymbol{W}_{emb} \cdot \boldsymbol{y}_t \qquad \boldsymbol{h}_t = \mathrm{RNN}(\boldsymbol{h}_{t-1}, \boldsymbol{v}_t) \qquad \hat{\boldsymbol{y}}'_{t+1} = \sigma(\boldsymbol{W}_o \cdot \boldsymbol{h}_t + \boldsymbol{b}_o)$$

$\boldsymbol{W}_{emb}, \boldsymbol{W}_o, \boldsymbol{b}_o$, and RNN's parameters are learned through SGD with loss function $\mathcal{L}$ defined by the binary cross entropy (BCE):

$$\mathcal{L} = \sum_{s \in \mathcal{D}} \sum_{t=1}^{T(s)-1} e(\boldsymbol{y}'_{t+1}, \hat{\boldsymbol{y}}'_{t+1}) \tag{36}$$

$$e(\boldsymbol{y}'_t, \hat{\boldsymbol{y}}'_t) = -[\boldsymbol{y}'_t \cdot \log \hat{\boldsymbol{y}}'_t + (\mathbf{1} - \boldsymbol{y}'_t) \cdot \log(\mathbf{1} - \hat{\boldsymbol{y}}'_t)] \tag{37}$$

where $\mathcal{D}$ is training set and $T(s)$ is length of a sequence s. This neural autoregressive approach has several benefits when modeling complex high-dimensional clinical sequences: First, low-dimensional embedding with $\boldsymbol{W}_{emb}$ helps us to obtain a compact representation of high-dimensional input vector $\boldsymbol{y}$. Second, complex dynamics of observed patient state sequences are modeled through RNN, which is capable of modeling non-linearities of the sequences. Furthermore, latent variables of neural models typically do not assume a specific probability form. Instead, the complex input-output association is learned through SGD based end-to-end learning framework, which allows more flexibility in modeling complex latent dynamics of observed sequence.

However, the neural autoregressive approach cannot address one important characteristic of the clinical sequence: the variability in the dynamics of sequences across different patients. Typically, EHR-derived clinical sequences consist of medical history of several tens of thousands of patients. The dynamics of one patient's sequence could be significantly different from the sequences of other patients. For typical neural autoregressive models, parameters of the trained model are used to process and predict sequences of *all* patients which consist of individual patients who can have different types of clinical complications, medication regimes, or observed sequence dynamics.

7.2.2 Subpopulation-based Online Model Adaptation

To address the patient variability issue, we propose a novel subpopulation-based learning framework that adapts the parameters of the neural autoregressive model to the past patients' sequences that are most similar to the current patient states. For simplicity, we denote population model ϕ^P as a model trained on all training set patient data $\mathcal{D}$, and subpopulation model ϕ^S as a model that is trained on a subset of training set data $\mathcal{D}^S$ that is close to the current patient state. Both models have identical model architecture.

Non-parametric Memory. The proposed learning framework is started by training ϕ^P with $\mathcal{D}$ and executing inference run for each time step $t' \in T(s‘)$ of all train set patients $s‘ \in \mathcal{D}$. Then we define a key-value pair $(k_{t‘}, v_{t‘})$ where the key is the hidden state vector $\boldsymbol{h}_{t'}$ and the value is the target event vector $\boldsymbol{y}_{t'+1}$. We store $(k_{t‘}, v_{t‘})$ into non-parametric storage (memory) $\mathcal{M}$:

$$\mathcal{M} = \{(\boldsymbol{h}_t, \boldsymbol{y}_{t+1}) | t \in T(s), s \in \mathcal{D}\} \tag{38}$$

Subpopulation Model Initialization. Then for each test set patient, we initialize ϕ^S with the parameters of ϕ^P to transfer general knowledge about *overall* patient state representation and dynamics to ϕ^S. However, due to patient variability issues, the parameters of ϕ^P could not be able to fully model the current patient's unique underlying clinical issues and status. Hence, its prediction can be limited to correctly predicting the future (next) clinical events.

Retrieval. We approach the issue mentioned above by adapting the parameters of ϕ^S with additional subpopulation data $\mathcal{D}^S$ which will be generated on the fly at each time step t of the current patient sequence. The subpopulation data $\mathcal{D}^S$ is retrieved from $\mathcal{M}$ as a k-nearest neighbors $\mathcal{N}$ of the current patient's hidden state $\boldsymbol{h}_t$ based on a distance function $d(\cdot, \cdot)$. In this study, we use L^2 distance function which is RBF kernel. The hidden state $\boldsymbol{h}_t$ is generated from population model ϕ^P. Since the similarity is calculated on the low-dimensional latent (hidden) state space defined by RNN, information from both the current input events $\boldsymbol{y}_t$ and the dynamics from the series of past events $\boldsymbol{y}_1 \dots \boldsymbol{y}_{t-1}$ is used to compute the similarity between current patient and $\mathcal{M}$:

$$\mathcal{D}^S = k\mathrm{NN}(\mathcal{M}, \boldsymbol{h}_t) \tag{39}$$

Subpopulation Model Adaptation. We adapt the parameters of ϕ^S first by computing an subpopulation error $\mathcal{L}^S = \sum_{(\boldsymbol{h}_i, \boldsymbol{y}_{i+1}) \in \mathcal{D}^S} e(\boldsymbol{y}_{i+1}, \phi^S(\boldsymbol{h}_i))$. Then, with $\mathcal{L}^S$ we iteratively update parameters of ϕ^S via SGD. Stopping criterion for the iterative update is: $\mathcal{L}^S(\tau - 1) - \mathcal{L}^S(\tau) < \epsilon$ where τ denotes the epoch of adaptation update and ϵ is a positive threshold.

7.2.3 Self-Adaptation Model

One limitation of the subpopulation-based model adaptation approach is that we still miss the chance to model the unique dynamics of the current patient's states and their specificity. To address this issue, we propose another novel learning framework that adapts the parameters of the neural autoregressive model to the current patient states based on the patient's past event sequence via SGD. We refer to this patient (instance) specific model as ϕ^I. As described in Algorithm 1, the online model adaptation procedure at time t for the current patient starts by creating a self-adaptation model ϕ^I from the population model ϕ^P. Similar to the subpopulation model, ϕ^I and ϕ^P have identical model architecture, and values of parameters in ϕ^I are initialized from ϕ^P to transfer the knowledge about general representation of patient states and their dynamics. Then, we compute an online patient-specific error $\mathcal{L}_t^I = \sum_{i=1}^{t-1} e(\boldsymbol{y}'_{i+1}, \hat{\boldsymbol{y}}'_{i+1})K(t,i)$ that reflects how much the prediction of ϕ^I deviates from the already observed target sequence for the current patient. With $\mathcal{L}_t^I$, we iteratively update parameters of ϕ^I via SGD. The same stopping criterion and training scheme of the subpopulation model is used here for the iterative update of ϕ^I.

Discounting. Please note that our adaptation-based loss $\mathcal{L}_t^I$ combines prediction errors for all time steps of the current patient's sequence. However, in order to better fit it to the most recent patient-specific behavior, we weigh the loss more towards recent clinical events. This is done by weighting prediction error for each step $i < t$ with $K(t,i)$ that is based on its time difference from the current time t. More specifically, $K(t,i)$ defines an exponential decay function:

$$K(t,i) = \exp\Big(-\frac{|t-i|}{\gamma}\Big) \tag{40}$$

where γ denotes the bandwidth (slope) of exponential decay; if γ is close to $+\infty$, errors at all time steps have the same weight.

Online Adaptation of Model Components. The RNN model may have too many parameters, and it may not help to adapt to all of them at the same time. One solution is to relax and permit to adapt only a subset of parameters. On the earlier work on self-adaptation model [112], three different settings for adapting parameters are experimented and compared: (a) output layer only ($\boldsymbol{W}_o, \boldsymbol{b}_o$), (b) transition model (RNN) only, and (c) combination of (a) and (b). From the experiment, (c) adapting only parameters of the output layer showed the best performance for predicting events. Based on this finding, we adapt the parameters of the output layer in this work.

Algorithm 1: Online Model Adaptation

```
Input  : Population model $\phi^P$, Current patient's history of **observed** input
         sequence $\boldsymbol{H}_t = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t\}$ and target sequence $(\boldsymbol{y}'_1, \ldots, \boldsymbol{y}'_t)$
Initialize self-adaptation model $\phi^I$ from $\phi^P$; $\tau = 0$; $\mathcal{L}_t^C(0) = \infty$;
repeat
    $\tau = \tau + 1$;
    $\mathcal{L}_t^C(\tau) = \sum_{i=1}^{t-1} e\big(\boldsymbol{y}'_{i+1}, \hat{\boldsymbol{y}}'_{i+1}\big) \cdot K(t,i)$ where $\hat{\boldsymbol{y}}'_{i+1} = \phi^I(\boldsymbol{H}_i)$;
    Update parameters of $\phi^I$ with $\mathcal{L}_t^C(\tau)$ via SGD;
until $\mathcal{L}_t^C(\tau-1) - \mathcal{L}_t^C(\tau) < \epsilon$;
Output: self-adaptation model $\phi^I$
```

7.2.4 Combined Adaptive Model

The common objective of the two (subpopulation and patient-specific) adaptation models is to represent better individual patient-specific behaviors and event sequences. Indeed, the two models learn different types of information from available patient event sequence data and they are complementary to each other. By learning from the small pool of most similar past patients' states and its outcome, the subpopulation model can cover dependencies between past and future events which are observed in a small group of patients with specific complications or diseases. On the other hand, the self-adaptation model learns unique dynamics and characteristics of the current patient's own past event sequence. Meanwhile, the best way to maximize the gain from the two different approaches is to unify the two methods, and the effective yet straightforward way to unify the two approaches is to combine the two losses $\mathcal{L}^S$ and $\mathcal{L}^I$ together:

$$\mathcal{L}^C = \mathcal{L}^I + \mu * \mathcal{L}^S \tag{41}$$

In this work, we have the combined adaptation model ϕ^C that is trained on $\mathcal{L}^C$ and report its performances along with the previous two approaches.

7.2.5 Meta Switching Mechanism

One limitation of the online adaptation approach is that it tries to modify the dynamics to fit more closely to the specifics of each patient's own sequence or other similar patients' sequences. However, when the patient's state changes suddenly due to recent events (e.g., a sudden clinical complication such as sepsis), the parameters of the adapted models ($\phi^{S,I,C}$) may not be able to adapt quickly enough to these changes. In such a case, switching back to a more general population model could be more desirable.

Model switching framework [129, 189] can resolve this issue by dynamically switching among a pool of available models such as subpopulation model ϕ^S, self-adaptation model ϕ^I, combined adapted model ϕ^C, and the population model ϕ^P. Driven by the recent performance of models, it can switch to the best performing model at each time step. Algorithm 2 implements the model switching idea. Given a trained population model ϕ^P, online adapted models $\phi^{S,I,C}$ trained via online adaptation, and the current patient's observed sequence, we can compute discounted losses $\mathcal{L}^{P,S,I,C}$ for these models on the past data. By comparing these losses, we select the model that gives the lowest error (averaged over $|E|$ event types) and use it for predicting the next step. We refer to prediction based on this meta switching mechanism as **meta-switching**.

A simple yet powerful extension of the meta switching mechanism is to allow selecting the best model for each event type (event-specific meta switching). One restriction of the aforementioned meta switching mechanism is that one best model is selected at each time step, and the model's prediction for the next step is used as the output of the meta switching mechanism. We relax this restriction by having *per event type* meta switching mechanism. For each event type, we select the best model among a pool of all available models based on each model's performance at the previous time step for each specific event type. This method is referred to as **meta-switching-event**.

```
Algorithm 2: Meta Model Switching
  Input  : $\phi^P$, $\phi^I$, $\phi^S$, $\phi^C$ $\boldsymbol{H}_t = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t\}$,$(\boldsymbol{y'}_1, \ldots, \boldsymbol{y'}_t)$
  $\mathcal{L}^I = \sum_{i=1}^t e(\boldsymbol{y'}_{i+1}, \hat{\boldsymbol{y'}}^I_{i+1}) \cdot K(t,i)$ where $\hat{\boldsymbol{y'}}^I_{i+1} = \phi^I(\boldsymbol{H}_i)$;
  $\mathcal{L}^P = \sum_{i=1}^t e(\boldsymbol{y'}_{i+1}, \hat{\boldsymbol{y'}}^P_{i+1}) \cdot K(t,i)$ where $\hat{\boldsymbol{y'}}^P_{i+1} = \phi^P(\boldsymbol{H}_i)$;
  $\mathcal{L}^S = \sum_{i=1}^t e(\boldsymbol{y'}_{i+1}, \hat{\boldsymbol{y'}}^S_{i+1}) \cdot K(t,i)$ where $\hat{\boldsymbol{y'}}^S_{i+1} = \phi^S(\boldsymbol{H}_i)$;
  $\mathcal{L}^C = \sum_{i=1}^t e(\boldsymbol{y'}_{i+1}, \hat{\boldsymbol{y'}}^C_{i+1}) \cdot K(t,i)$ where $\hat{\boldsymbol{y'}}^C_{i+1} = \phi^C(\boldsymbol{H}_i)$;
  $\hat{\boldsymbol{y'}}_{t+1} = \hat{\boldsymbol{y'}}^z_{t+1}$ where $z = \arg\min_{z \in \{I,P,S,C\}} \left(\mathcal{L}^z\right)$
  Output: Prediction at time step $t+1$: $\hat{\boldsymbol{y'}}_{t+1}$
```

7.3 Experimental Evaluation

7.3.1 Experiment Setup

In this section, we evaluate the performance of the proposed models, including patient-specific adaptation model (**PatSpecAdap**), sub-population adaptation model (**Subpop-Adap**), and combined adaptation model (**CombinedAdap**) model as well as meta-switching mechanism (**Meta-Switch**) and its event-specific extension (**Meta-Switch-Event**) on the real-world EHRs data in compare it with baseline models. Likewise, most details of the experimental setup (including clinical data, feature preparation, and evaluation metrics) are identical to the experiment setup introduced in Section 3.3 of Chapter 3. For kernel bandwidth γ, we use fixed value 3.0.

7.3.2 Results on Personalized Adaptive Models vs. Population Model

We first compare the prediction performance of the population model (GRU-POP) and the proposed methods on different adaptation mechanisms: subpopulation-based adaptation (SubpopAdap), patient-specific adaptation (PatSpecAdap), and combined adaptation (CombinedAdap) which uses both subpopulation and patient-specific instances for personalized model adaptation. As shown in Figure 38, the combined adaptation model and patient-specific adaptation model clearly outperform the population-based model across most of the time steps. Especially on earlier days of admissions (day=1-3), the self-adaptation model performs better than the population model with a decent margin. But subpopulation model underperforms the population model, and it also affects the combined model's performance on the first time step (day). But as time progresses, the overall performance gap between the combined model and the population model increases. On day 19, while the self-adaptation model's performance is almost the same as the population model, the combined model's performance is significantly better than population model's performance with the help of information from the subpopulation model. That is, we can see that when subpopulation model is solely used, it underperforms than population model overall. This is somehow expected as the parameters of subpopulation model are *indirectly* tuned (adapted) to the current patient through k-nearest neighbor retrieval of other similar patients from the training set data. Therefore, the specificity of the current patient's underlying states is not directly modeled into the parameters of subpopulation model. Nonetheless, the benefit of subpopulation model is revealed through the competency of the combined model. That is, the improved performance of combined model compared to patient specific model can be understood as the additional information provided through the subpopulation model.

7.3.3 Results for Meta Switching Mechanism

We also experiment with meta online switching approach. It chooses the best predictive model from among a pool of available prediction models. We run the method to choose among the population-based model ϕ^P and different adaptation models based on subpopulation ϕ^S, patient-specific sequence ϕ^I, and combined approach ϕ^C.

As shown in Figure 39, models that rely on multiple models and online switching clearly outperform baseline models of GRU-POP, CNN, and RETAIN. In particular, the event-specific extension of the meta switching mechanism (Meta-Switch-Event) greatly surpasses the prediction performances of all other models. This shows flexibility in selecting the best model for each event type at each time step substantially benefits the task of predicting complex multivariate clinical event sequences consisting of heterogeneous individual event time series with different temporal characteristics and dependencies to precursor events.

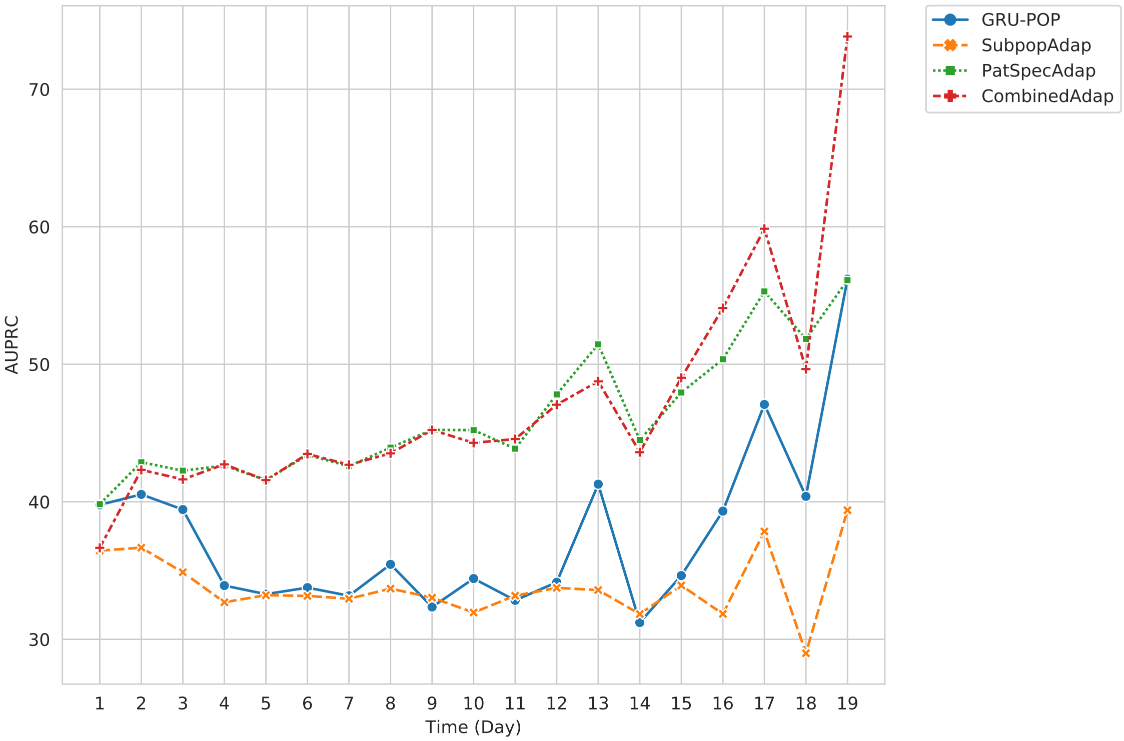


Figure 38: Prediction performance (AUPRC) of the population-based model (GRU-POP) and proposed personalized models based on different mechanisms: subpopulation-based adaptation (SubpopAdap), patient-specific adaptation (PatSpecAdap), and combined adaptation (CombinedAdap) which uses both subpopulation and patient-specific instances for personalized online adaptation.

model	AUPRC
CNN	35.85
RETAIN	39.22
GRU-POP	36.37
SubpopAdap	32.19
PatSpecAdap	39.67
CombinedAdap	39.06
Meta-Switch	41.35
Meta-Switch-Event	57.81

Table 6: Prediction results (AUPRC) of all models. Personalized Meta-Switch-Event outperforms population model (GRU-POP) by 58.9% AUPRC gain.

When the prediction performance is averaged across all time steps, we can observe that the event-specific meta-switching mechanism outperforms all models, as shown in Table 6. Particularly, the event-specific meta switching mechanism's AUPRC is +58.9% higher than the population model. The non-event-specific version of meta switching increases AUPRC by 13.6% from the population model. These results reveal the distinct advantage added by the meta online switching methods.

7.3.3.1 When the Model Switches?

To better understand the behavior of online meta switching-based adaptation, we investigate when the model switches to each model among a pool of available models, including the subpopulation model, self-adaptation model, combined model, and population model. First, we analyze the proportion of how many times each model is used at each time step across all test-set patient sequences from the meta-switching mechanism. As shown in Figure 40, in the first time step, the population model is used 28%, and the subpopulation model is used 14%. Then, subsequently, the usage ratio of the two models drastically decreased, and the self-adaptation model and the combined model are mostly used in later time steps. Especially although the direct ratio of the subpopulation model is, in general, low, its contribution can be found in the fact that the combined model is dominantly used across most time steps (day 2 through day 15). Around the end of the time steps (day 16 through day 19), the ratio for the self-adaptation model is increased. This can be explained by the fact that self-adaptation model can have enough observations to adapt the patient-specific variability in that latter time of sequences and can provide the best prediction among the pool of all available models. To properly interpret the results, Figure 41 shows the number of patients in each time step. This number can also be interpreted as the length of patient sequences and their volume. We can clearly see that the number of patients with longer sequences is minimal, as the majority of sequences are very short. For example, patients with sequences longer than 13 days of admission are only about 12% of all patients in the test set. We can conclude that the population model is often biased towards the dynamics and characteristics of shorter patient sequences. Meanwhile, proposed online adaptation models can effectively learn and adapt better to the dynamics of longer sequences.

7.3.3.2 Predicting Repetitive and Non-Repetitive Events

To perform this analysis, we divide event occurrences into two groups based on whether the same type of event has or has not occurred before. We compute AUPRC for each group as shown in Table 7. The results show that for **non-repetitive events**, the performance of the self-adaptation model is the lowest among all models. This is expected because, with no previous occurrence of a target event, the self-adaptation model could have difficulty making an accurate prediction for the new target event. In this case, we can also see the benefit of the online switching mechanism: the prediction of the population model is more accurate than the self-adaptation model, and the online switching mechanism correctly chooses the population model. More specifically, the Meta-Switch mechanism recovers most of the predictability of GRU-POP for non-repetitive event prediction. For **repetitive event prediction**, we can see that both population models and personalized adapted models have similar performances. However, the online switching approaches (Meta-Switch and Meta-Switch-Event) are the best and outperform all other approaches.

7.3.3.3 Event-type-specific Performance

We also examine the performance of the online meta switching model (Meta-Switch-Event) compared to the population model (GRU-POP) at the individual event level. Specifically, for each event type, we compute two statistics: first, we compute the percentage difference (%+) between the two models, and then we compute each event type's occurrence rate in all possible time windows (W=24), averaged across all test set patient sequences. Then, we plot the two statistics in a scatter plot as shown in Figure 42. Even though the correlation coefficient is weak (-0.24), we can see those event types that have larger performance gaps (e.g., $>$ 100%+) are indeed less occurring events (e.g., occurrence rate $<$ 0.1). This also reveals that our proposed approaches effectively improve prediction performance, especially for events with smaller data points. It is a valuable characteristic for clinical event time series prediction where data are usually scarce.

7.3.3.4 Results based on Event Categories

We analyze the experimental results further by breaking the evaluation results down by inspecting the performances of the models for the four different event categories: medication administration events, lab test events, physiological events, and procedure events. For all $|E|$=282 target event types, we averaged prediction performances of them based on the four-event categories. The results are shown in Figure 43. The proposed methods (Combined Adaptation model, Meta switch mechanism, and Event-specific meta switch mechanism) consistently outperform baseline models across all event categories in AUPRC statistics over all time-steps. Especially, the results of the event-specific meta switch mechanism (Meta-Switch-Event) are on par.

Notably, the performance gap between proposed personalized models (e.g., Meta-Switch-Event) and population model is larger for the event categories with overall predictability is relatively lower such as medication administration or lab test results. On the other hand, in the physiological signal category where most models' performance is high, the performance gap between the proposed method and the population model is much smaller. Consistent with the results shown in the event-type-specific result in Section 7.3.3.3, this shows the proposed personalized approach has a unique competency to improve the prediction performance more where the population model, in general, performs worse.

7.4 Summary

This chapter has developed methods for patient-specific adaptation of predictive models of clinical event sequences. We proposed two novel event time series prediction solutions that attempt to adjust the predictions for individual patients through an online model update. We demonstrate the improved performance of our models through experiments on MIMIC-3, a publicly available dataset of electronic health records for ICU patients, and show significant improvement in next event occurrence prediction performance.

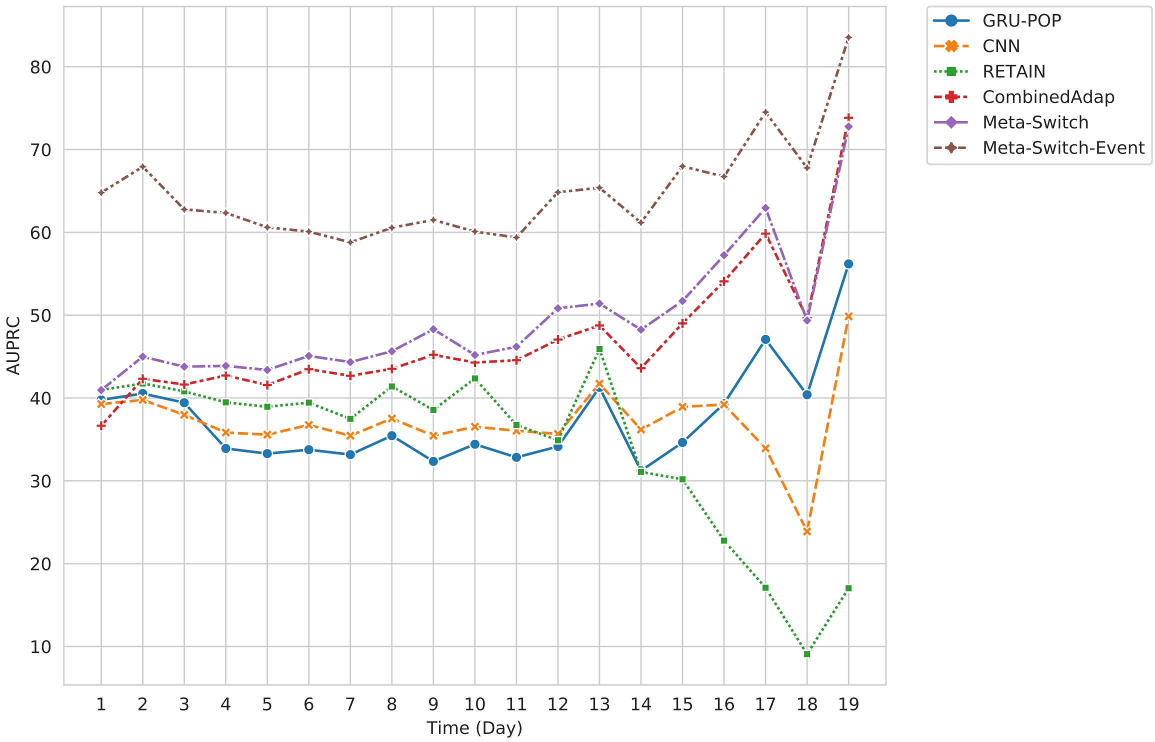
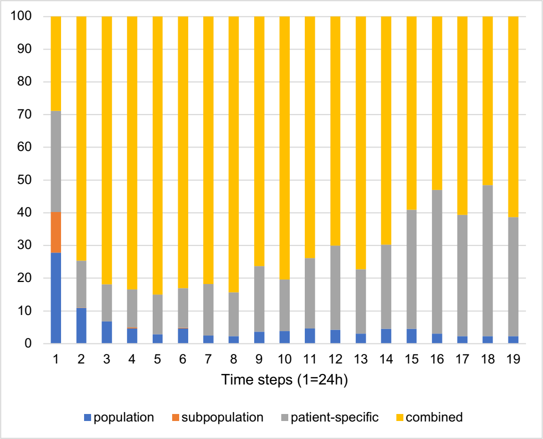


Figure 39: Performance of meta online switching method (Meta-Switch) with population and patient-specific adaptation models, and its extension to event-specific switching mechanism (Meta-Switch-Event). Meta online switching methods clearly outperform all baseline models (GRU-POP, RETAIN, CNN)



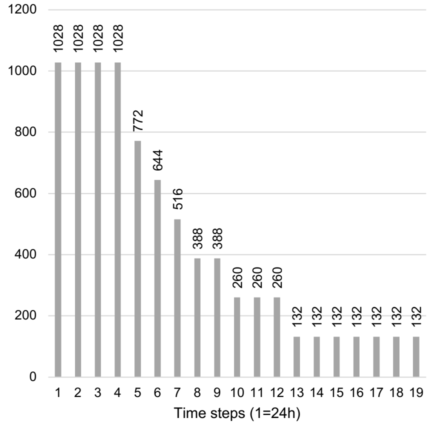


Figure 40: Ratio of different models selected in Meta-Switch mechanism. Population model's ratio is quickly decreasing as proposed personalized models provide better predictability.

Figure 41: Number of test set patients in each time step. The number of patients quickly deteriorates with longer sequence lengths.

	Non-repetitive	Repetitive
CNN	15.28	46.87
RETAIN	16.60	47.34
GRU-POP	15.94	47.28
SubpopAdap	14.07	45.91
PatSpecAdap	13.00	48.17
CombinedAdap	14.52	47.15
Meta-Switch	16.60	49.82
Meta-Switch-Event	40.42	66.68

Table 7: Prediction result on non-repetitive and repetitive event groups. For non-repetitive events, the performance of the self-adaptation model is the lowest. However, the online switching approaches (Meta-Switch, Meta-Switch-Event) recover the predictability by switching to the population model and show the best performance across both groups.

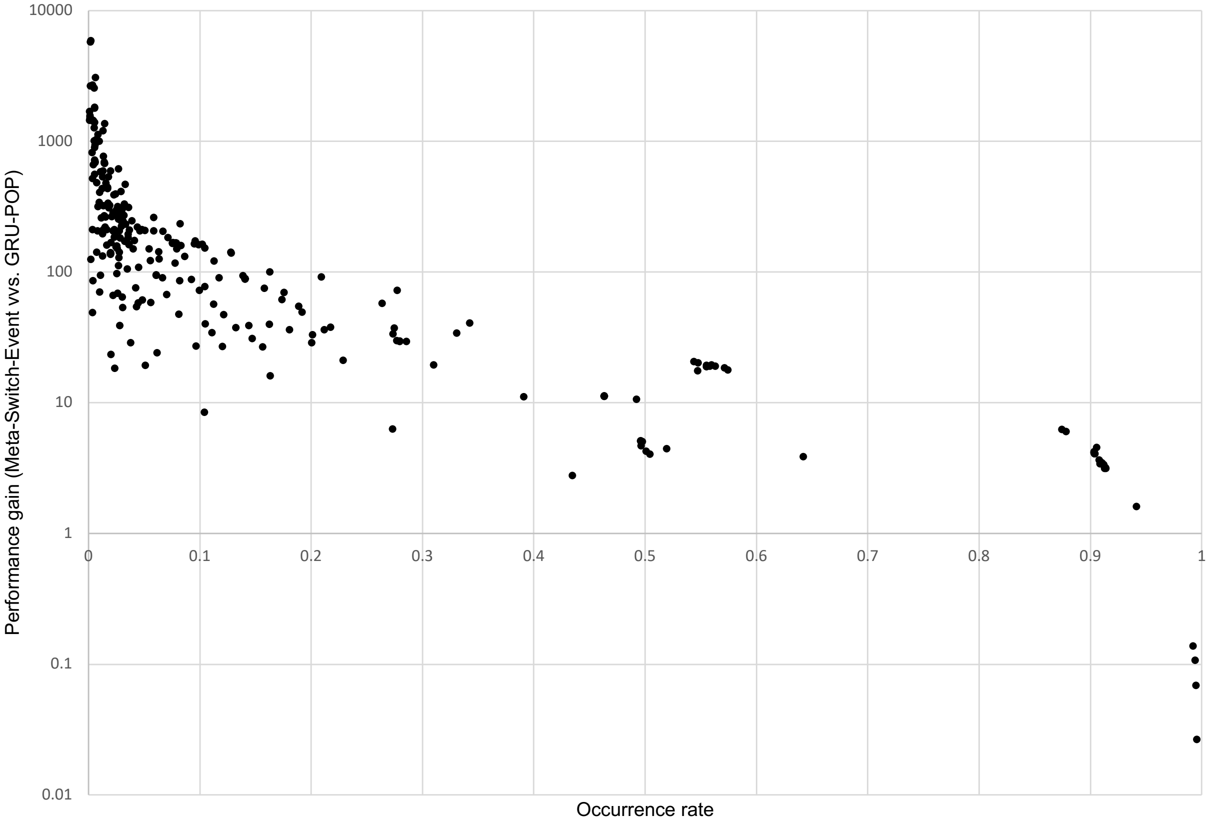
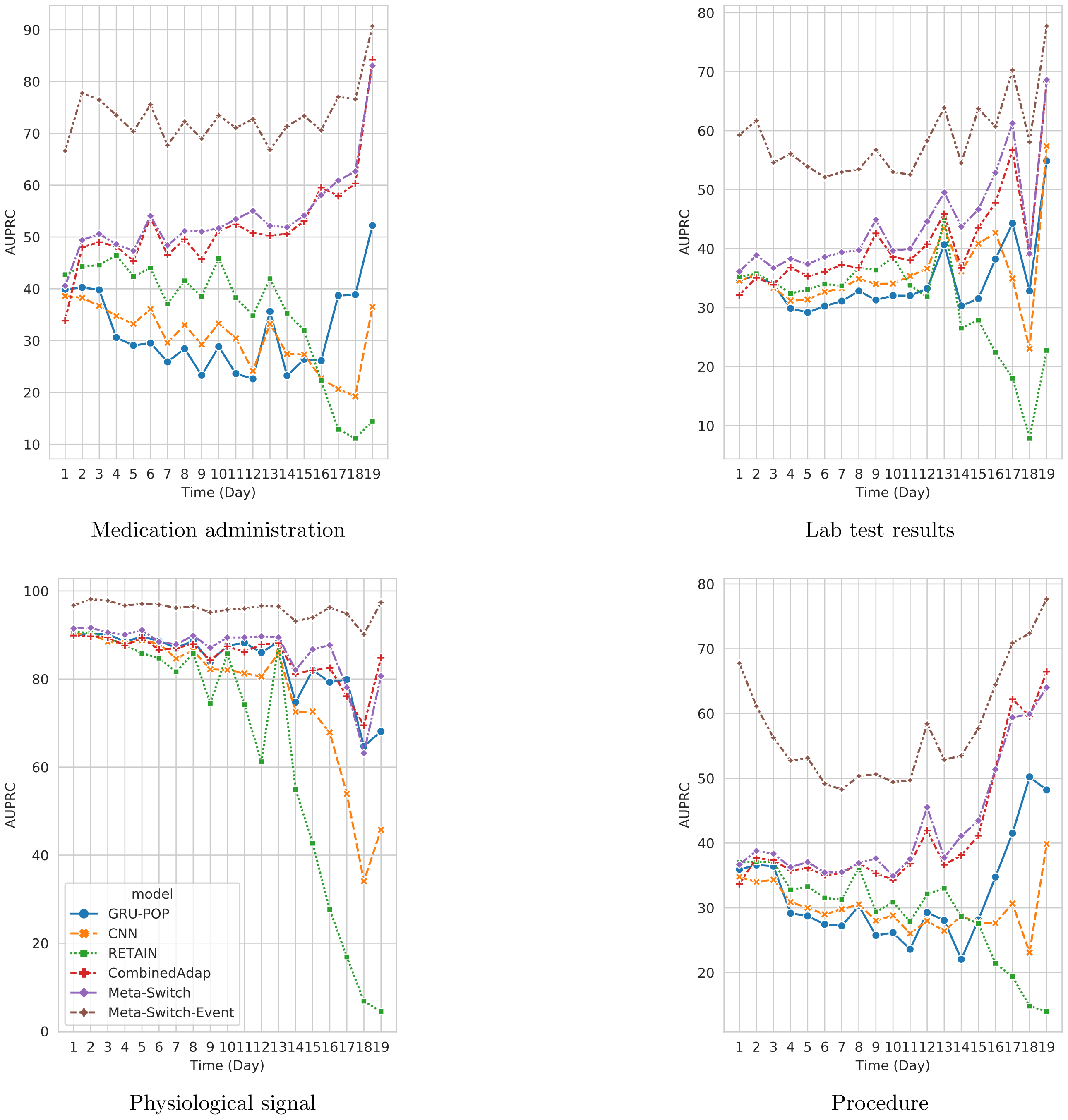


Figure 42: Scatter plot on performance difference between the population model (GRU-POP) and online meta switching-based adaptation model (Meta-Switch-Event) and occurrence rate of each event type.



Medication administration

Lab test results

Physiological signal

Procedure

Figure 43: Prediction results by the event type category

8.0 Conclusion

In this dissertation, we studied neural temporal models that can predict future occurrences of the multivariate clinical event time series. We presented several methods that address two overarching questions: (1) how to learn effective patient state representation and transitions with characteristics of EHRs-derived clinical event time series and (2) how to learn personalized and adaptive dynamic state representation that can address the variability of heterogeneous patient event sequences. The main contributions of this dissertation are summarized below.

- In Chapter 3, we developed recent context-aware LSTM model. We hypothesized that information on recently occurred events could provide strong predictability toward the next event occurrence. To properly model information from both recent and long-term past events, we developed a new event time series model based on the long-short-term-memory (LSTM) that relies on two sources of information to predict future events. One source is derived from the set of recently observed clinical events. The other one is based on the hidden state space defined by the LSTM that aims to abstract past, more distant, patient information that is predictive of future events. In the context of Markov state models, the next state in our models and the transition to the next state is defined by a combination of the recent state (most recent events) and the hidden state summarizing more distant past events.
- In Chapter 4, we proposed a new method for modeling Dependencies on periodically occurring events in clinical event time series data. We hypothesized that (1) many events in the EHR-based multivariate event time series occur periodically and (2) proper modeling of the periodically occurring events could increase the predictability toward the next event prediction. To overcome the limitations of RNN/LSTM-based approach to modeling periodicity of the time series, we proposed a novel yet simple mechanism to enhance the handling of periodic events and incorporate them into the prediction. We developed an external memory that stores observed temporal characteristics of many periodic events and uses them to derive a new periodicity-aware signal to further enhance

event predictions, and this at any time, and for any prediction window size. The external memory store gaps (time differences) observed for pairs of two consecutive events of the same type (a) for all past patients and (b) for the current patient. At the time of the prediction, the proposed model calculates how much time has elapsed since the latest occurrence of the event of the same type, and based on the prediction window size and information stored in the memory of past event gaps, it predicts the probability of the signal to be repeated in the next prediction window.

- In Chapter 5, we presented a new neural memory module called Multi-scale Temporal Memory (MTM) linking events in a distant past with the current prediction time. Through a novel mechanism implemented in MTM, information about previous events on different time scales is compiled and read on-the-fly for prediction through memory contents. We demonstrated the efficacy of MTM by combining it with different patient state summarization methods that cover different temporal aspects of patient states.
- In Chapter 6, we developed a specialized neural sequence model (RNN) based on the Mixture-of-Experts (MoE) architecture. We hypothesized that clinical event sequences in EHRs are generated from a pool of heterogeneous patients where each patient typically has different types of complications. While average behaviors of clinical event sequences could be captured by a single model, the dynamics of heterogeneous event sequences could not be well captured by a single model. The heterogeneity of various patient sequences is modeled through multiple experts that consist of Gated Recurrent Unit (GRU). Particularly, instead of directly training MoE from scratch, we augment MoE based on the prediction signal from the pretrained base GRU model. With this way, the mixture of experts can provide flexible adaptation to the (limited) predictive power of the single base GRU model.
- In Chapter 7, we presented two novel event time series prediction solutions that attempt to better adapt the population models to the individual patient. First, we proposed a model that aims to improve a prediction made for the current patient at any specific time using a repository of event sequences recorded for past patients. The model works by first identifying the patient states among past patients that are most similar to the current state of the current patient and then adapting the predictions of the population-wide

model with the help of outcomes recorded for such patients and their states. We refer to this model as the *subpopulation model*. Second, we developed and studied a model that adapts the predictions of the population-wide model only based on the patients' own sequence. We refer to this model as the *self-adaptation model*. However, one concern with either the sub-population or the self-adaptation model and related adaptation is that it may lose some flexibility by being fit too tightly to the specific patient (and patient's recent condition) or to the patient state most similar to the current state. To address this, we also developed and investigated the *meta-switching framework* that is able to dynamically identify and switch to the best model to follow for the current patient. Briefly, the meta-framework uses a set of models and learns how to dynamically identify and adaptively switch to the model offering the most promising solution. Such a framework may combine the population, subpopulation, and self-adaptation models. We note, that all of the above solutions can extend RNN-based multivariate sequence prediction to support personalized clinical event sequence adaptation.

However, the methods for clinical event time series prediction proposed in this thesis come with some limitations and challenges:

- **Modeling in Discrete Time.** In this thesis, we investigated event time series methods for discretized time. That is, we discretize the time series with a fixed-size window (e.g., 6 hours) and consider the events in the same time window that occurred at the same time. Although this approach has benefits such as computational efficiency, we lose information about accurate event timings of historical events and target events, and their dependencies with time. With methods based on Hawkes Processes and Point Processes, we could build event time series models in continuous time. Developing efficient continuous-time events representation learning and transition methods is an important research topic.

  Modeling these data in multi-modalities can improve the understanding of the underlying patient state as well as predictions for future events. Especially since the event prediction models in this thesis are based on deep neural networks, we can use powerful neural network architectures for computer vision (e.g., ResNet, VGG) or text (e.g., BERT) data and incorporate them into event prediction models. Then, we train the parameters of these models together on a single objective function for event prediction.

- **Multi-Modal Clinical Data.** While EHR consists of various data in different modalities such as images (e.g., X-Ray and MRI) or text (e.g., clinical notes written by physicians and nurses), the focus of this thesis is on the recordings of clinical events and their attributes in EHR. The data in various modalities can provide complementary information about patient states beyond our current event time-series-based approach. For example, clinical notes contain a rich summarization of the past and current patient conditions, clinical actions, and prognosis. On the other hand, clinical imaging data provide additional salient information about specific physical conditions. Our proposed approaches in the thesis miss this opportunity to model and utilize the rich multi-modal data.
- **Deployment of Personalized Adaptive Models.** In this thesis, we proposed novel personalized adaptation methods that build prediction models that are specific to individual patient's dynamic states. To be deployed and used in real-world hospitals, these models need to be regularly retrained to adapt their parameters to dynamically changing patient conditions. The too short time interval between two consecutive recurring training sessions may cause instability in the model parameter. The long interval may fail to capture details of patient dynamics and deter such models' efficacy for predictive care. Additionally, since we create and train individual models for each patient, we need to have a sufficient scalable computing infrastructure to be able to serve thousands of patients in real-time concurrently.

Meanwhile, the limitations and challenges of the proposed methods may open up new research opportunities and directions. In the following, we briefly outline some of these opportunities.

- **Self-Supervised (Contrastive) Learning**. One major challenge of modeling event-time series derived from EHR-derived clinical data is the data scarcity of the long-tail (less-occurring) events. In this thesis, we tackle this issue by developing multiple information channels where each channel addresses a different aspect of event time series (e.g., modeling repetitively occurring events with the method presented in Chapter 4). Another interesting approach to treat this issue is to employ self-supervised learning. By randomly augmenting input events and training the model with contrastive loss function (objective function promotes input data augmented pair of the same instance and demotes a pair of random different instances), we can create a more robust model for less-occurring target events. Another benefit of this approach is that we can explore this approach on top of the methods presented in this thesis since it can be applied to any type of model.

- **Learning from Soft Labels**. Typical setup for training machine learning models for event predictions is that events are modeled as binary variables, that is, a predicted event is either absent or present. However, some events, especially those that are based on various rule-based definitions of conditions or outcomes may come with a great deal of uncertainty related to the definition of the concept itself. Take for example, the concept of hypotension. Should a patient whose diastolic blood pressure is 60 mm Hg or lower considered to be hypotensive, and the patient with diastolic pressure 62 mm Hg as not hypotensive. In reality some of the concepts and their definitions are blurred and uncertain. In these cases the learning of predictive models can be improved by considering uncertainty of the concept (event) and its occurrence and training the model using this information. This is the main idea of training the classification model with *soft label information* which has demonstrated improved learning of classification models especially in cases when prior of the concept occurrence is very low. Pioneered in the work [160, 161, 162] and extended to active learning settings in [218, 219, 220, 221]. The gist of the approach is to consider and take into account the soft labels for training the prediction model and use ordering/ranking solution to define the models that respect this order as much as possible.
- **Knowledge-Guided Learning**. One distinct characteristic of the clinical domain compared to other machine learning fields (e.g., computer vision, NLP) is that human knowledge and expertise of the field is condensed and available in the form of textbooks and knowledge bases (ontologies). Clinical terminology and their relationships are available in Unified Medical Language System (UMLS) knowledge bases such as ICD9CM (In-

ternational Classification of Diseases) which classifies diseases and their diagnosis and procedures, NDFRT (National Drug File - Reference Terminology) which classifies drugs and their ingredients, structures, diseases it treats, and LOINC (Logical Observation Identifiers Names and Codes) that contains information about lab tests and its related concepts such as disease and chemical components, etc. The extensive clinical knowledge and practices are written in clinical textbooks and are often available also online. One possible direction for enhancing modeling of the clinical event time series derived from EHRs is to utilize the human knowledge when modeling low-prior events, and when the the sample sizes are not sufficient to reliably infer the relations among the events.

Appendix

A.1 Statistical Significance Test

We conduct statistical significance test (paired T-Test) for the models we proposed in each chapter. As shown in the Table 8, all proposed models in each chapter show statistically significant improvement over corresponding baseline models.

Chapter	Baseline vs. Proposed model	Window size	T-Test P-value	P-value ≤ 0.05?
3	HS vs. HS-RC	24	1.336E-11	True
		12	2.18346E-09	True
		6	4.08392E-07	True
4	HS-RC vs. HS-RC-PM	24	2.2078E-09	True
		12	1.63409E-07	True
		6	4.45064E-12	True
5	HS-RC-PM vs. HS-RC-PM-MTM	24	6.08409E-09	True
		12	0.00084682	True
		6	9.68268E-09	True
6	RETAIN vs. R-MOE	24	4.18043E-11	True
7	GRU-POP vs. Meta-Switch-Event	24	3.99423E-16	True
	GRU-POP vs. Meta-Switch	24	4.33602E-13	True
	GRU-POP vs. CombinedAdap	24	1.35466E-09	True
	GRU-POP vs. SelfAdap	24	2.67492E-10	True

Table 8: Statistical significance test (Paired T-Test) results for models proposed in each chapter

Bibliography

[1] Eytan Adar, Jaime Teevan, and Susan T Dumais. Large scale analysis of web revisitation patterns. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 1197–1206. ACM, 2008.

[2] Sharmin Afrose, Wenjia Song, Charles B Nemeroff, Chang Lu, and Danfeng Daphne Yao. Subpopulation-specific machine learning prognosis for underrepresented patients with double prioritized bias correction. *medRxiv*, 2021.

[3] Denis Agniel, Isaac S Kohane, and Griffin M Weber. Biases in electronic health record data due to processes within the healthcare system: retrospective observational study. *Bmj*, 361, 2018.

[4] Adebiyi A Ariyo, Adewumi O Adewumi, and Charles K Ayo. Stock price prediction using the arima model. In *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, pages 106–112. IEEE, 2014.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[6] Jacek M. Bajor and Thomas A. Lasko. Predicting medications from diagnostic codes with recurrent neural networks. In *ICLR*, 2017.

[7] Noam Barda, Gal Yona, Guy N Rothblum, Philip Greenland, Morton Leibowitz, Ran Balicer, Eitan Bachmat, and Noa Dagan. Addressing bias in prediction models by improving subpopulation calibration. *Journal of the American Medical Informatics Association*, 28(3):549–558, 2021.

[8] Matthew Barren and Milos Hauskrecht. Improving prediction of low-prior clinical events with simultaneous general patient-state representation learning. In *International Conference on Artificial Intelligence in Medicine*, pages 479–490. Springer, 2021.

[9] Iyad Batal, Gregory Cooper, and Milos Hauskrecht. A Bayesian Scoring Technique for Mining Predictive and Non-Spurious Rules. In *Proceedings of the European conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, 2012.

[10] Iyad Batal, Gregory F Cooper, Dmitriy Fradkin, James Harrison, Fabian Moerchen, and Milos Hauskrecht. An efficient pattern mining approach for event detection in multivariate temporal data. *Knowledge and information systems*, 46(1):115–150, 2016.

[11] Iyad Batal, Dmitriy Fradkin, James Harrison, Fabian Moerchen, and Milos Hauskrecht. Mining Recent Temporal Patterns for Event Detection in Multivariate Time Series Data. In *Proceedings of the international conference on Knowledge Discovery and Data mining (SIGKDD)*, 2012.

[12] Iyad Batal and Milos Hauskrecht. A supervised time series feature extraction technique using dct and dwt. In *2009 International Conference on Machine Learning and Applications*, pages 735–739, 2009.

[13] Iyad Batal and Milos Hauskrecht. A Concise Representation of Association Rules using Minimal Predictive Rules. In *Proceedings of the European conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, 2010.

[14] Iyad Batal, Lucia Sacchi, Riccardo Bellazzi, and Milos Hauskrecht. Multivariate time series classification with temporal abstractions. In *Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference, FLAIRS-22*, pages 344–349. University of Pittsburgh, 2009.

[15] Iyad Batal, Lucia Sacchi, Riccardo Bellazzi, and Milos Hauskrecht. A temporal abstraction framework for classifying clinical temporal data. In *AMIA Annual Symposium Proceedings*, volume 2009, page 29. American Medical Informatics Association, 2009.

[16] Iyad Batal, Hamed Valizadegan, Gregory F. Cooper, and Milos Hauskrecht. A pattern mining approach for classifying multivariate temporal data. In *Proceedings of the IEEE international conference on bioinformatics and biomedicine (BIBM)*, 2011.

[17] Iyad Batal, Hamed Valizadegan, Gregory F Cooper, and Milos Hauskrecht. A temporal pattern mining approach for classifying electronic health record data. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(4):1–22, 2013.

[18] Leonard Baum. An inequality and associated maximization technique in statistical estimation of probabilistic functions of a markov process. *Inequalities*, 3:1–8, 1972.

[19] Inci M Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K Jain, and Jiayu Zhou. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 65–74, 2017.

[20] J Robert Beck and Stephen G Pauker. The markov process in medical prognosis. *Medical decision making*, 3(4):419–458, 1983.

[21] Rinaldo Bellomo, Claudio Ronco, John A Kellum, Ravindra L Mehta, Paul Palevsky, et al. Acute renal failure–definition, outcome measures, animal models, fluid therapy and information technology needs: the second international consensus conference of the acute dialysis quality initiative (adqi) group. *Critical care*, 8(4):R204, 2004.

[22] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[23] Christos Berberidis, Walid G Aref, Mikhail Atallah, Ioannis Vlahavas, Ahmed K Elmagarmid, et al. Multiple and partial periodicity mining in time series databases. In *ECAI*, volume 2, pages 370–374, 2002.

[24] Carlo Berzuini, Riccardo Bellazzi, Silvana Quaglini, and D.J. Spiegelhalter. Bayesian networks for patient monitoring. *Artificial Intelligence in Medicine*, 4:243–260, 05 1992.

[25] Ingi orleifur Bjarnason. Earthquake sequence 1973–1996 in bárarbunga volcano: Seismic activity leading up to eruptions in the nw-vatnajökull area. 2014.

[26] David Blumenthal and John P Glaser. Information technology comes to medicine. *The New England journal of medicine*, 356(24):2527–2534, 2007.

[27] Byron Boots. Learning stable linear dynamical systems. *Online]. Avail.: https://www. ml. cmu. edu/research/dap-papers/dap_boots. pdf*, 2009.

[28] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[29] Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*, 2017.

[30] Ao Buke, Fang Gaoli, Wang Yongcai, Song Lei, and Yang Zhiqi. Healthcare algorithms by wearable inertial sensors: a survey. *China Communications*, 12(4):1–12, 2015.

[31] Basit Chaudhry, Jerome Wang, Shinyi Wu, Margaret Maglione, Walter Mojica, Elizabeth Roth, Sally C Morton, and Paul G Shekelle. Systematic review: impact of health information technology on quality, efficiency, and costs of medical care. *Annals of internal medicine*, 144(10):742–752, 2006.

[32] Pin-An Chen, Li-Chiu Chang, and Fi-John Chang. Reinforced recurrent neural networks for multi-step-ahead flood forecasts. *Journal of Hydrology*, 497:71–79, 2013.

[33] Li-Fang Cheng, Bianca Dumitrascu, Gregory Darnell, Corey Chivers, Michael Draugelis, Kai Li, and Barbara E Engelhardt. Sparse multi-output gaussian processes for online medical time series prediction. *BMC medical informatics and decision making*, 20(1):1–23, 2020.

[34] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[35] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Doctor AI: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*, pages 301–318, 2016.

[36] Edward Choi, Mohammad Taha Bahadori, Elizabeth Searles, Catherine Coffey, Michael Thompson, James Bost, Javier Tejedor-Sojo, and Jimeng Sun. Multi-layer representation learning for medical concepts. In *The 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1495–1504. ACM, 2016.

[37] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. GRAM: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 787–795, 2017.

[38] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. RETAIN: An interpretable predictive model for healthcare using

reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pages 3504–3512, 2016.

[39] Edward Choi, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Using recurrent neural network models for early detection of heart failure onset. *Journal of the American Medical Informatics Association*, 24(2):361–370, 2017.

[40] Sharath Cholleti, Andrew Post, Jingjing Gao, Xia Lin, William Bornstein, Dedra Cantrell, and Joel Saltz. Leveraging derived data elements in data analytic models for understanding and predicting hospital readmissions. In *AMIA Annual Symposium Proceedings*, volume 2012, page 103. American Medical Informatics Association, 2012.

[41] Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. Hierarchical multiscale recurrent neural networks. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[42] Jenny Cifuentes, Geovanny Marulanda, Antonio Bello, and Javier Reneses. Air temperature forecasting using machine learning techniques: a review. *Energies*, 13(16):4215, 2020.

[43] AR De Kruyk, JH van der Meulen, LA Van Herwerden, JA Bekkers, EW Steyerberg, R Dekker, and JD Habbema. Use of markov series and monte carlo simulation in predicting replacement valve performances. *The Journal of heart valve disease*, 7(1):4–12, 1998.

[44] Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge. Transformers4Rec: Bridging the gap between NLP and sequential / session-based recommendation. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*, pages 143–153. ACM, 2021.

[45] Arthur P Dempster. Maximum likelihood estimation from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 39:1–38, 1977.

[46] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[47] Gianfranco Doretto, Alessandro Chiuso, Ying Nian Wu, and Stefano Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003.

[48] Finale Doshi-Velez, Yaorong Ge, and Isaac Kohane. Comorbidity clusters in autism spectrum disorders: an electronic health record time-series analysis. *Pediatrics*, 133(1):e54–e63, 2014.

[49] Olga Ésik, Gábor Tusnády, Lajos Trón, András Boér, Zoltán Szentirmay, István Szabolcs, Károly Rácz, Erzsébet Lengyel, Judit Székely, and Miklós Kásler. Markov model-based estimation of individual survival probability for medullary thyroid cancer patients. *Pathology Oncology Research*, 8(2):93–104, 2002.

[50] C. Esteban, D. Schmidt, D. Krompaß, and V. Tresp. Predicting sequences of clinical events by using a personalized temporal latent embedding model. pages 130–139, October 2015.

[51] Cristóbal Esteban, Oliver Staeck, Stephan Baier, Yinchong Yang, and Volker Tresp. Predicting clinical events by combining static and dynamic information using recurrent neural networks. In *2016 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 93–101. Ieee, 2016.

[52] Yuchen Fan, Yao Qian, Feng-Long Xie, and Frank K Soong. Tts synthesis with bidirectional lstm based recurrent neural networks. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[53] Wael Farhan, Zhimu Wang, Yingxiang Huang, Shuang Wang, Fei Wang, and Xiaoqian Jiang. A predictive model for medical events based on contextual embedding of temporal sequences. *JMIR medical informatics*, 4(4):e39, 2016.

[54] Anthony T Fojo, Katherine L Musliner, Peter P Zandi, and Scott L Zeger. A precision medicine approach for psychiatric disease based on repeated symptom scores. *Journal of psychiatric research*, 95:147–155, 2017.

[55] Abdur Rahim Mohammad Forkan and Ibrahim Khalil. A probabilistic model for early prediction of abnormal clinical events using vital sign correlations in home-based monitoring. In *2016 IEEE international conference on pervasive computing and communications (PerCom)*, pages 1–9. IEEE, 2016.

[56] Zoubin Ghahramani and Geoffrey E Hinton. Parameter estimation for linear dynamical systems. Technical report, Technical Report CRG-TR-96-2, University of Totronto, Dept. of Computer Science, 1996.

[57] Shameek Ghosh, Jinyan Li, Longbing Cao, and Kotagiri Ramamohanarao. Septic shock prediction for ICU patients via coupled HMM walking on sequential contrast patterns. *Journal of biomedical informatics*, 66:19–31, 2017.

[58] Jennifer C Ginestra, Heather M Giannini, William D Schweickert, Laurie Meadows, Michael J Lynch, Kimberly Pavan, Corey J Chivers, Michael Draugelis, Patrick J Donnelly, Barry D Fuchs, et al. Clinician perception of a machine learning-based early warning system designed to predict severe sepsis and septic shock. *Critical care medicine*, 47(11):1477, 2019.

[59] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

[60] Prosanta Gope and Tzonelih Hwang. Bsn-care: A secure iot-based modern healthcare system using body sensor network. *IEEE sensors journal*, 16(5):1368–1376, 2015.

[61] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, pages 1764–1772, 2014.

[62] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.

[63] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

[64] Nooshin HajiGhassemi and Marc Deisenroth. Analytic long-term forecasting with periodic gaussian processes. In *Artificial Intelligence and Statistics*, pages 303–311, 2014.

[65] Robert M Hamer and Pippa M Simpson. Last observation carried forward versus mixed models in the analysis of psychiatric clinical trials, 2009.

[66] Min Han, Jianhui Xi, Shiguo Xu, and Fu-Liang Yin. Prediction of chaotic time series based on the recurrent predictor neural network. *IEEE transactions on signal processing*, 52(12):3409–3416, 2004.

[67] Md Rafiul Hassan and Baikunth Nath. Stock market forecasting using hidden Markov model: a new approach. In *Intelligent Systems Design and Applications, 2005. ISDA'05. Proceedings. 5th International Conference on*, pages 192–196. IEEE, 2005.

[68] Moeen Hassanalieragh, Alex Page, Tolga Soyata, Gaurav Sharma, Mehmet Aktas, Gonzalo Mateos, Burak Kantarci, and Silvana Andreescu. Health monitoring and management using internet-of-things (iot) sensing with cloud-based processing: Opportunities and challenges. In *2015 IEEE International Conference on Services Computing*, pages 285–292. IEEE, 2015.

[69] Milos Hauskrecht, Iyad Batal, Charmgil Hong, Quang Nguyen, Gregory F Cooper, Shyam Visweswaran, and Gilles Clermont. Outlier-based detection of unusual patient-management actions: an icu study. *Journal of biomedical informatics*, 64:211–221, 2016.

[70] Milos Hauskrecht, Iyad Batal, Michal Valko, Shyam Visweswaran, Gregory F Cooper, and Gilles Clermont. Outlier detection for patient monitoring and alerting. *Journal of biomedical informatics*, 46(1):47–55, 2013.

[71] Milos Hauskrecht, Michal Valko, Iyad Batal, Gilles Clermont, Shyam Visweswaran, and Gregory F Cooper. Conditional outlier detection for clinical alerting. In *AMIA annual symposium proceedings*, volume 2010, page 286. American Medical Informatics Association, 2010.

[72] Milos Hauskrecht, Michal Valko, Branislav Kveton, Shyam Visweswaran, and Gregory F Cooper. Evidence-based anomaly detection in clinical domains. In *AMIA Annual Symposium Proceedings*, volume 2007, page 319. American Medical Informatics Association, 2007.

[73] Katharine E Henry, David N Hager, Peter J Pronovost, and Suchi Saria. A targeted real-time early warning score (trewscore) for septic shock. *Science translational medicine*, 7(299):299ra122–299ra122, 2015.

[74] Abram Hindle, Michael W Godfrey, and Richard C Holt. Mining recurrent activities: Fourier analysis of change events. In *2009 31st International Conference on Software Engineering-Companion Volume*, pages 295–298. IEEE, 2009.

[75] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(2):107–116, 1998.

[76] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[77] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[78] Sandy H Huang, Paea LePendu, Srinivasan V Iyer, Ming Tai-Seale, David Carrell, and Nigam H Shah. Toward personalizing treatment for depression: predicting diagnosis and severity. *Journal of the American Medical Informatics Association*, 21(6):1069–1075, 2014.

[79] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[80] Richard Hughey and Anders Krogh. Hidden Markov models for sequence analysis: extension and analysis of the basic method. *Bioinformatics*, 12(2):95–107, 1996.

[81] Kyuyeon Hwang and Wonyong Sung. Character-level language modeling with hierarchical recurrent neural networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5720–5724. IEEE, 2017.

[82] Oliver Ibe. *Markov processes for stochastic modeling.* Newnes, 2013.

[83] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[84] Martin Jacobsen. *Point process theory and applications: marked point and piecewise deterministic processes.* Springer Science & Business Media, 2006.

[85] Abhyuday N Jagannatha and Hong Yu. Bidirectional rnn for medical event detection in electronic health records. In *Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting*, volume 2016, page 473. NIH Public Access, 2016.

[86] Abhyuday N Jagannatha and Hong Yu. Structured prediction models for rnn based sequence labeling in clinical text. In *Proceedings of the conference on empirical methods in natural language processing. conference on empirical methods in natural language processing*, volume 2016, page 856. NIH Public Access, 2016.

[87] Ashish K Jha, Catherine M DesRoches, Eric G Campbell, Karen Donelan, Sowmya R Rao, Timothy G Ferris, Alexandra Shields, Sara Rosenbaum, and David Blumenthal. Use of electronic health records in us hospitals. *New England Journal of Medicine*, 360(16):1628–1638, 2009.

[88] Tanvi Jindal, Prasanna Giridhar, Lu-An Tang, Jun Li, and Jiawei Han. Spatiotemporal periodical pattern mining in traffic data. In *Proceedings of the 2nd ACM SIGKDD international workshop on urban computing*, page 11. ACM, 2013.

[89] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[90] Rudolf Emil Kalman. Mathematical description of linear dynamical systems. *Journal of the Society for Industrial and Applied Mathematics*, 1(2):152–192, 1963.

[91] Komal Kapoor, Karthik Subbian, Jaideep Srivastava, and Paul Schrater. Just in time recommendations: Modeling the dynamics of boredom in activity streams. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 233–242. ACM, 2015.

[92] Elnaz Karimi. *Integrative Predictive Support Systems for Hospital's Resource Planning and Scheduling.* PhD thesis, Ecole Polytechnique, Montreal (Canada), 2018.

[93] Tohru Katayama. *Subspace methods for system identification.* Springer Science & Business Media, 2006.

[94] AKI Kdigo. Work group. kdigo clinical practice guideline for acute kidney injury. *Kidney Int Suppl*, 2(1):1–138, 2012.

[95] John A Kellum and Azra Bihorac. Artificial intelligence to predict aki: is it a breakthrough? *Nature Reviews Nephrology*, pages 1–2, 2019.

[96] Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. Structured attention networks. *arXiv preprint arXiv:1702.00887*, 2017.

[97] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[98] J. F. C. Kingman. *Poisson processes*, volume 3 of *Oxford Studies in Probability*. The Clarendon Press Oxford University Press, New York, 1993. Oxford Science Publications.

[99] Roopa Kohli-Seth and John M Oropello. The future of bedside monitoring. *Critical care clinics*, 16(4):557–578, 2000.

[100] Takeshi Kurashima, Tim Althoff, and Jure Leskovec. Modeling interdependent and periodic real-world action sequences. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 803–812. International World Wide Web Conferences Steering Committee, 2018.

[101] Thomas A Lasko, Joshua C Denny, and Mia A Levy. Computational phenotype discovery using unsupervised feature learning over noisy, sparse, and irregular clinical data. *PloS one*, 8(6):e66341, 2013.

[102] Günter Last and Andreas Brandt. *Marked Point Processes on the real line: the dynamical approach*. Springer Science & Business Media, 1995.

[103] Günter Last and Mathew Penrose. *Lectures on the Poisson process*, volume 7. Cambridge University Press, 2017.

[104] Patrick J Laub, Thomas Taimre, and Philip K Pollett. Hawkes processes. *arXiv preprint arXiv:1507.02822*, 2015.

[105] Michael Lawton, Fahd Baig, Michal Rolinski, Claudio Ruffman, Kannan Nithi, Margaret T May, Yoav Ben-Shlomo, and Michele Hu. Parkinson's disease subtypes in the oxford parkinson disease centre (opdc) discovery cohort. *Journal of Parkinson's disease*, 5(2):269–279, 2015.

[106] Jae Won Lee. Stock price prediction using reinforcement learning. In *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No. 01TH8570)*, volume 1, pages 690–695. IEEE, 2001.

[107] JaeHo Lee. Smart health: Concepts and status of ubiquitous health with smartphone. In *ICTC 2011*, pages 388–389. IEEE, 2011.

[108] Jeong Min Lee and Milos Hauskrecht. Recent-context-aware lstm-based clinical time-series prediction. In *In Proceedings of AI in Medicine Europe (AIME)*, 2019.

[109] Jeong Min Lee and Milos Hauskrecht. Clinical event time-series modeling with periodic events. In *The Thirty-Third International FLAIRS Conference*. AAAI, 2020.

[110] Jeong Min Lee and Milos Hauskrecht. Multi-scale temporal memory for clinical event time-series prediction. In *2020 International Conference on Artificial Intelligence in Medicine (AIME 2020)*, 2020.

[111] Jeong Min Lee and Milos Hauskrecht. Modeling multivariate clinical event time-series with recurrent temporal mechanisms. *Artificial Intelligence in Medicine*, 112:102021, 2021.

[112] Jeong Min Lee and Milos Hauskrecht. Neural clinical event sequence prediction through personalized online adaptive learning. In *19th International Conference on Artificial Intelligence in Medicine (AIME 2021)*, pages https–arxiv. https://link.springer.com/chapter/10.1007%2F978-3-030-77211-6_20, 2021.

[113] Jeong Min Lee and Milos Hauskrecht. Learning to adapt clinical sequences with residual mixture of experts. In *2022 International Conference on Artificial Intelligence in Medicine (AIME 2022)*, pages https–arxiv, 2022.

[114] Jeong Min Lee and Aldrian Obaja Muis. Diagnosis code prediction from electronic health records as multilabel text classification: a survey. *URL: http://people. cs. pitt. edu/˜ jlee/papers/cp1_survey_jlee_amuis. pdf [accessed 2021-05-09]*, 2021.

[115] Jeongmin Lee, James P McCusker, and Deborah L McGuinness. Climate change, disaster and sentiment analysis over social media mining. In *AGU Fall Meeting Abstracts*, volume 2012, pages IN51C–1703, 2012.

[116] SJG Lewis, Thomas Foltynie, Andrew D Blackwell, Trevor W Robbins, Adrian M Owen, and Roger A Barker. Heterogeneity of parkinson's disease in the early clinical

stages using a data driven approach. *Journal of Neurology, Neurosurgery & Psychiatry*, 76(3):343–348, 2005.

[117] Yikuan Li, Shishir Rao, Jose Roberto Ayala Solares, Abdelaali Hassaine, Rema Ramakrishnan, Dexter Canoy, Yajie Zhu, Kazem Rahimi, and Gholamreza Salimi-Khorshidi. Behrt: transformer for electronic health records. *Scientific reports*, 10(1):1–12, 2020.

[118] Rui Lin, Shujie Liu, Muyun Yang, Mu Li, Ming Zhou, and Sheng Li. Hierarchical recurrent neural network for document modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 899–907, 2015.

[119] Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[120] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2):212–261, February 1994.

[121] Ming Liu and Younghoon Kim. Classification of heart diseases based on ecg signals using long short-term memory. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2707–2710. IEEE, 2018.

[122] Siqi Liu and Milos Hauskrecht. Nonparametric regressive point processes based on conditional gaussian processes. In *Advances in Neural Information Processing Systems*, pages 1062–1072, 2019.

[123] Siqi Liu and Milos Hauskrecht. Event outlier detection in continuous time. In *Proceedings of the 38th International Conference on Machine Learning, PMLR 139*, 2021.

[124] Siqi Liu and Milos Hauskrecht. Event outlier detection in continuous time. In *International Conference on Machine Learning*, pages 6793–6803. PMLR, 2021.

[125] Zitao Liu and Milos Hauskrecht. Clinical time series prediction: Toward a hierarchical dynamical system framework. *Artificial Intelligence in Medicine*, 65(1):5–18, 2015.

[126] Zitao Liu and Milos Hauskrecht. A regularized linear dynamical system framework for multivariate time series analysis. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 1798–1804, 2015.

[127] Zitao Liu and Milos Hauskrecht. Learning adaptive forecasting models from irregularly sampled multivariate clinical data. In *The 30th AAAI Conference on Artificial Intelligence*, pages 1273–1279, 2016.

[128] Zitao Liu and Milos Hauskrecht. Learning linear dynamical systems from multivariate time series: A matrix factorization based framework. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 810–818. SIAM, 2016.

[129] Zitao Liu and Milos Hauskrecht. A personalized predictive framework for multivariate clinical time series via adaptive model selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1169–1177, 2017.

[130] Zitao Liu, Lei Wu, and Milos Hauskrecht. Modeling clinical time series using gaussian process sequences. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 623–631. SIAM, 2013.

[131] Hong Liu-Seifert, Shuyu Zhang, Deborah D'Souza, and Vladimir Skljarevski. A closer look at the baseline-observation-carried-forward (bocf). *Patient preference and adherence*, 4:11, 2010.

[132] L Ljung. System identification-theory for the user 2nd edition ptr prentice-hall. *Upper Saddle River, NJ*, 1999.

[133] Tsung-Chien Lu, Chia-Ming Fu, Matthew Huei-Ming Ma, Cheng-Chung Fang, and Anne M Turner. Healthcare applications of smart watches. *Applied clinical informatics*, 7(3):850–869, 2016.

[134] Duc Thanh Anh Luong and Varun Chandola. A k-means approach to clustering disease progressions. In *2017 IEEE International conference on healthcare informatics (ICHI)*, pages 268–274. IEEE, 2017.

[135] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[136] Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1903–1911, 2017.

[137] Iain L MacDonald and Walter Zucchini. *Hidden Markov and other models for discrete-valued time series*, volume 110. CRC Press, 1997.

[138] Salim Malakouti and Milos Hauskrecht. Hierarchical adaptive multi-task learning framework for patient diagnoses and diagnostic category classification.

[139] Salim Malakouti and Milos Hauskrecht. Not all samples are equal: Class dependent hierarchical multi-task learning for patient diagnosis classification. In *The Thirty-Third International Flairs Conference*, 2020.

[140] Seyedsalim Malakouti and Milos Hauskrecht. Predicting patient's diagnoses and diagnostic categories from clinical-events in ehr data. In *Conference on Artificial Intelligence in Medicine in Europe*, pages 125–130. Springer, 2019.

[141] MM Malik, S Abdallah, and M Ala'raj. Data mining and predictive analytics applications for the delivery of healthcare services: a systematic literature review. *Annals of Operations Research*, 270(1):287–312, 2018.

[142] Eric J Manders and Benoit M Dawant. Data acquisition for an intelligent bedside monitoring system. In *Proceedings of 18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 5, pages 1987–1988. IEEE, 1996.

[143] Subramani Mani, Yukun Chen, Tom Elasy, Warren Clayton, and Joshua Denny. Type 2 diabetes risk forecasting from emr data using machine learning. In *AMIA annual symposium proceedings*, volume 2012, page 606. American Medical Informatics Association, 2012.

[144] Subramani Mani, Asli Ozdas, Constantin Aliferis, Huseyin Atakan Varol, Qingxia Chen, Randy Carnevale, Yukun Chen, Joann Romano-Keeler, Hui Nian, and Jörn-Hendrik Weitkamp. Medical decision support using machine learning for early detection of late-onset neonatal sepsis. *Journal of the American Medical Informatics Association*, 21(2):326–336, 2014.

[145] Matteo Mantovani, Carlo Combi, and Milos Hauskrecht. Mining compact predictive pattern sets using classification model. In *Conference on Artificial Intelligence in Medicine in Europe*, pages 386–396. Springer, 2019.

[146] Yaroslav Marchuk, Rudys Magrans, Bernat Sales, Jaume Montanya, Josefina López-Aguilar, Candelaria De Haro, Gemma Gomà, Carles Subirà, Rafael Fernández, Robert M Kacmarek, et al. Predicting patient-ventilator asynchronies with hidden markov models. *Scientific reports*, 8(1):1–7, 2018.

[147] Joao Maroco, Dina Silva, Ana Rodrigues, Manuela Guerreiro, Isabel Santana, and Alexandre de Mendonça. Data mining methods in the prediction of dementia: A real-data comparison of the accuracy, sensitivity and specificity of linear discriminant analysis, logistic regression, neural networks, support vector machines, classification trees and random forests. *BMC research notes*, 4(1):1–14, 2011.

[148] James P McCusker, Deborah L McGuinness, Jeongmin Lee, Chavon Thomas, Paul Courtney, Zaria Tatalovich, Noshir Contractor, Glen Morgan, and Abdul Shaikh. Towards next generation health data exploration: a data cube-based investigation into population statistics for tobacco. In *2013 46th Hawaii International Conference on System Sciences*, pages 2725–2732. IEEE, 2013.

[149] Jim McCusker, Jeongmin Lee, Chavon Thomas, and Deborah L McGuinness. Public health surveillance using global health explorer. In *SATBI+ SWIM*, 2012.

[150] Eddie McKenzie. Ch. 16. discrete variate time series. *Handbook of Statistics*, 21:573–606, 12 2003.

[151] Ravindra L Mehta, John A Kellum, Sudhir V Shah, Bruce A Molitoris, Claudio Ronco, David G Warnock, Adeera Levin, et al. Acute kidney injury network: report of an initiative to improve outcomes in acute kidney injury. *Critical care*, 11(2):R31, 2007.

[152] Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, pages 6754–6764, 2017.

[153] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[154] Frank J Molnar, Brian Hutton, and Dean Fergusson. Does analysis using "last observation carried forward" introduce bias in dementia research? *Cmaj*, 179(8):751–753, 2008.

[155] Sarah Mullin, Jaroslaw Zola, Robert Lee, Jinwei Hu, Brianne MacKenzie, Arlen Brickman, Gabriel Anaya, Shyamashree Sinha, Angie Li, and Peter L Elkin. Longitudinal k-means approaches to clustering and analyzing ehr opioid use trajectories for clinical subtypes. *Journal of Biomedical Informatics*, 122:103889, 2021.

[156] Jyoti R Munavalli, Shyam Vasudeva Rao, Aravind Srinivasan, Usha Manjunath, and GG Van Merode. A robust predictive resource planning under demand uncertainty to improve waiting times in outpatient clinics. *Journal of health management*, 19(4):563–583, 2017.

[157] Shamim Nemati, Andre Holder, Fereshteh Razmi, Matthew D Stanley, Gari D Clifford, and Timothy G Buchman. An interpretable machine learning model for accurate prediction of sepsis in the icu. *Critical care medicine*, 46(4):547, 2018.

[158] Phuoc Nguyen, Truyen Tran, and Svetha Venkatesh. Finding algebraic structure of care in time: A deep learning approach. *ArXiv*, abs/1711.07980, 2017.

[159] Phuoc Nguyen, Truyen Tran, Nilmini Wickramasinghe, and Svetha Venkatesh. Deepr: a convolutional net for medical records. *IEEE journal of biomedical and health informatics*, 21(1):22–30, 2016.

[160] Quang Nguyen, Hamed Valizadegan, and Milos Hauskrecht. Learning classification with auxiliary probabilistic information. In *IEEE International Conference on Data Mining*, pages 477–486, 2011.

[161] Quang Nguyen, Hamed Valizadegan, and Milos Hauskrecht. Sample-efficient learning with auxiliary class-label information. In *Proceedings of the Annual American Medical Informatics Association Symposium*, pages 1004–1012, 2011.

[162] Quang Nguyen, Hamed Valizadegan, and Milos Hauskrecht. Learning classification models with soft-label information. *Journal of American Medical Informatics Association*, 2013.

[163] Diogo Nunes, Teresa Rocha, Vicente Traver, C Teixeira, M Ruano, Simão Paredes, P Carvalho, and Jorge Henriques. Latent states extraction through kalman filter for the prediction of heart failure decompensation events. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3947–3950. IEEE, 2019.

[164] Michael A Osborne, Stephen J Roberts, Alex Rogers, Sarvapali D Ramchurn, and Nicholas R Jennings. Towards real-time information processing of sensor network data using computationally efficient multi-output gaussian processes. In *2008 International Conference on Information Processing in Sensor Networks (ipsn 2008)*, pages 109–120. IEEE, 2008.

[165] Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*, 2016.

[166] Yubin Park and Joydeep Ghosh. A hierarchical ensemble of α-trees for predicting expensive hospital visits. In *International Conference on Brain Informatics and Health*, pages 178–187. Springer, 2014.

[167] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

[168] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[169] Trang Pham, Truyen Tran, Dinh Phung, and Svetha Venkatesh. Predicting healthcare trajectories from medical records: A deep learning approach. *Journal of biomedical informatics*, 69:218–229, 2017.

[170] Bart Post, Johannes D Speelman, and Rob J De Haan. Clinical heterogeneity in newly diagnosed parkinson's disease. *Journal of neurology*, 255(5):716–722, 2008.

[171] Sara Rabhi. *Optimized deep learning-based multimodal method for irregular medical timestamped data.* PhD thesis, Institut Polytechnique de Paris, 2022.

[172] Sara Rabhi, Ronay Ak, Gabriel de Souza Pereira Moreira, Jeong Min Lee, and Even Oldridge. Effectiveness of transformers on session-based recommendation. In *16th Women in Machine Learning Workshop (WiML) in NeurIPS 2021*, 2021.

[173] Sara Rabhi, Jérémie Jakubowicz, and Marie-Helene Metzger. Deep learning versus conventional machine learning for detection of healthcare-associated infections in french clinical narratives. *Methods of information in medicine*, 58(01):031–041, 2019.

[174] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[175] Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M Dai, Nissan Hajaj, Michaela Hardt, Peter J Liu, Xiaobing Liu, Jake Marcus, Mimi Sun, et al. Scalable and accurate deep learning with electronic health records. *NPJ Digital Medicine*, 1(1):18, 2018.

[176] Daniel Ramage. Hidden markov models fundamentals. *CS229 Section Notes*, 1, 2007.

[177] Shishir Rao, Yikuan Li, Rema Ramakrishnan, Abdelaali Hassaine, Dexter Canoy, John Cleland, Thomas Lukasiewicz, Gholamreza Salimi-Khorshidi, and Kazem Rahimi. An explainable transformer-based deep learning model for the prediction of incident heart failure. *arXiv preprint arXiv:2101.11359*, 2021.

[178] Jakob Gulddahl Rasmussen. Lecture notes: Temporal point processes and the conditional intensity function. *arXiv preprint arXiv:1806.00221*, 2018.

[179] Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):1–13, 2021.

[180] Marian-Andrei Rizoiu, Young Lee, Swapnil Mishra, and Lexing Xie. A tutorial on hawkes processes for events in social media. *arXiv preprint arXiv:1708.06401*, 2017.

[181] Dimitris Rizopoulos. Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics*, 67(3):819–829, 2011.

[182] Mark S Roberts. Markov process-based monte carlo simulation: a tool for modeling complex disease and its application to the timing of liver transplantation. In *Proceedings of the 24th conference on winter simulation*, pages 1034–1040, 1992.

[183] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than ROC plot when evaluating binary classifiers on imbalanced datasets. *PloS One*, 10(3), 2015.

[184] Peter Schulam and Suchi Saria. A framework for individualizing predictions of disease trajectories by exploiting multi-resolution structure. In *Advances in NeurIPS*, pages 748–756, 2015.

[185] Peter Schulam, Fredrick Wigley, and Suchi Saria. Clustering longitudinal clinical marker trajectories from electronic health data: Applications to phenotyping and

endotype discovery. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[186] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[187] Howard D Sesso, Roland S Chen, Gilbert J L'Italien, Pablo Lapuerta, Won Chan Lee, and Robert J Glynn. Blood pressure lowering and life expectancy based on a markov model of cardiovascular events. *Hypertension*, 42(5):885–890, 2003.

[188] Kristen A. Severson, Lana M. Chahine, Luba Smolensky, Kenney Ng, Jianying Hu, and Soumya Ghosh. Personalized input-output hidden markov models for disease progression modeling. In Finale Doshi-Velez, Jim Fackler, Ken Jung, David Kale, Rajesh Ranganath, Byron Wallace, and Jenna Wiens, editors, *Proceedings of the 5th Machine Learning for Healthcare Conference*, volume 126 of *Proceedings of Machine Learning Research*, pages 309–330. PMLR, 07–08 Aug 2020.

[189] Shai Shalev-Shwartz et al. Online learning and online convex optimization. 2011.

[190] Anis Sharafoddini, Joel A Dubin, and Joon Lee. Finding similar patient subpopulations in the icu using laboratory test ordering patterns. In *Proceedings of the 2018 7th International Conference on Bioinformatics and Biomedical Science*, pages 72–77, 2018.

[191] Anis Sharafoddini, Joel A Dubin, David M Maslove, Joon Lee, et al. A new insight into missing data in intensive care unit patient profiles: observational study. *JMIR medical informatics*, 7(1):e11605, 2019.

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

[193] Benjamin A Smallheer. Technology and monitoring patients at the bedside. *Nursing Clinics*, 50(2):257–268, 2015.

[194] Padhraic Smyth. Clustering sequences with hidden Markov models. In *Advances in neural information processing systems*, pages 648–654, 1997.

[195] Huan Song, Deepta Rajan, Jayaraman J Thiagarajan, and Andreas Spanias. Attend and diagnose: Clinical time series analysis using attention models. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[196] Jingkuan Song, Zhao Guo, Lianli Gao, Wu Liu, Dongxiang Zhang, and Heng Tao Shen. Hierarchical lstm with adjusted temporal attention for video captioning. *arXiv preprint arXiv:1706.01231*, 2017.

[197] Ruslan Leont'evich Stratonovich. Conditional Markov processes. *Theory of Probability & Its Applications*, 5(2):156–178, 1960.

[198] Ilya Sutskever. *Training recurrent neural networks.* University of Toronto Toronto, Ontario, Canada, 2013.

[199] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[200] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[201] Shakirah Mohd Taib, Azuraliza Abu Bakar, Abdul Razak Hamdan, and SM Syed Abdullah. Classifying weather time series using featurebased approach. *Int. J. Adv. Soft Comput. Its Appl*, 7(3), 2015.

[202] William Trouleau, Azin Ashkan, Weicong Ding, and Brian Eriksson. Just one more: Modeling binge watching behavior. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1215–1224. ACM, 2016.

[203] Srinivasan Vairavan, Larry Eshelman, Syed Haider, Abigail Flower, and Adam Seiver. Prediction of mortality in an intensive care unit using logistic regression and a hidden markov model. In *2012 Computing in Cardiology*, pages 393–396. IEEE, 2012.

[204] Hamed Valizadegan, Quang Nguyen, and Milos Hauskrecht. Learning classification models from multiple experts. *Journal of Biomedical Informatics*, pages 1125–1135, 2013.

[205] Michal Valko and Milos Hauskrecht. Feature importance analysis for patient management decisions. *Studies in health technology and informatics*, 160(Pt 2):861, 2010.

[206] Peter Van Overschee and BL De Moor. *Subspace identification for linear systems: Theory—Implementation—Applications*. Springer Science & Business Media, 2012.

[207] Stephanie M Van Rooden, Willem J Heiser, Joost N Kok, Dagmar Verbaan, Jacobus J Van Hilten, and Johan Marinus. The identification of parkinson's disease subtypes using cluster analysis: a systematic review. *Movement disorders*, 25(8):969–978, 2010.

[208] Sandeep Kumar Vashist, E Marion Schneider, and John HT Luong. Commercial smartphone-based devices and smart applications for personalized healthcare monitoring and management. *Diagnostics*, 4(3):104–128, 2014.

[209] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[210] Shyam Visweswaran and Gregory F Cooper. Instance-specific bayesian model averaging for classification. In *Advances in Neural Information Processing Systems*, pages 1449–1456, 2005.

[211] Andrew J Viterbi. A personal history of the viterbi algorithm. *IEEE Signal Processing Magazine*, 23(4):120–142, 2006.

[212] Michail Vlachos, Philip Yu, and Vittorio Castelli. On periodicity detection and structural periodic similarity. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 449–460. SIAM, 2005.

[213] Paul J Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356, 1988.

[214] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[215] Ronald J Williams and David Zipser. Gradient-based learning algorithms for recurrent. *Backpropagation: Theory, architectures, and applications*, 433, 1995.

[216] Diane Wong, Reshma Modi, and Murali Ramanathan. Assessment of markov-dependent stochastic models for drug administration compliance. *Clinical pharmacokinetics*, 42(2):193–204, 2003.

[217] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural im-

age caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[218] Yanbing Xue and Milos Hauskrecht. Active learning of classification models with likert-scale feedback. In *SIAM International Conference on Data Mining (SDM)*, pages 28–35, 2017.

[219] Yanbing Xue and Milos Hauskrecht. Efficient learning of classification models from soft-label information by binning and ranking. In *Proceedings of the 30th International Florida AI Research Society Conference*, pages 164–169, 2017.

[220] Yanbing Xue and Milos Hauskrecht. Active learning of multi-class classifiers with auxiliary probabilistic information. In *Proceedings of the 31st International Florida AI Research Society Conference.*, 2018.

[221] Yanbing Xue and Milos Hauskrecht. Active learning of multi-class classification models from ordered class sets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5589–5596, 2019.

[222] Pranjul Yadav, Michael Steinbach, Vipin Kumar, and Gyorgy Simon. Mining electronic health records (ehrs) a survey. *ACM Computing Surveys (CSUR)*, 50(6):1–40, 2018.

[223] Chin-Sheng Yang, Chih-Ping Wei, Chi-Chuan Yuan, and Jen-Yu Schoung. Predicting the length of hospital stay of burn patients: Comparisons of prediction accuracy among different clinical stages. *Decision Support Systems*, 50(1):325–335, 2010.

[224] Ke Yu, Mingda Zhang, Tianyi Cui, and Milos Hauskrecht. Monitoring icu mortality risk with a long short-term memory recurrent neural network. In *PACIFIC SYMPOSIUM ON BIOCOMPUTING 2020*, pages 103–114. World Scientific, 2019.

[225] Quan Yuan, Jingbo Shang, Xin Cao, Chao Zhang, Xinhe Geng, and Jiawei Han. Detecting multiple periods and periodic patterns in event time sequences. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 617–626. ACM, 2017.

[226] Xi Zhang, Jingyuan Chou, Jian Liang, Cao Xiao, Yize Zhao, Harini Sarva, Claire Henchcliffe, and Fei Wang. Data-driven subtyping of parkinson's disease using longitudinal clinical records: a cohort study. *Scientific reports*, 9(1):1–12, 2019.

[227] Yutao Zhang, Robert Chen, Jie Tang, Walter F Stewart, and Jimeng Sun. Leap: learning to prescribe effective and safe treatment combinations for multimorbidity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1315–1324. ACM, 2017.

[228] Bin Zhao, Xuelong Li, and Xiaoqiang Lu. HSA-RNN: Hierarchical structure-adaptive rnn for video summarization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.