POLITECNICO MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Executive Summary of the Thesis

Generation of Synthetic Data From Digital Health Records

Laurea Magistrale in Computer Science and Engineering - Ingegneria Informatica

Author: Tancredi Covioli

Advisor: Prof. Marco Gribaudo

Co-advisor: Dott. Ing. Tommaso Dolci

Academic year: 2021-2022

1. Introduction

Big data has flooded the healthcare field with the introduction of new tools for continuous patient monitoring, producing massive amounts of structured and unstructured data every day. For this reason, medical facilities are moving towards data-driven health-care, with the objective of leveraging this data to support clinical decision-making and public health management. This new request to elaborate a multitude of heterogeneous data has resulted in the emergence of new systems capable of processing it as well as the need to evaluate their performance.
One of the techniques commonly used for the evaluation of such Systems is modeling, which consists in performing the evaluation on a model of the system under analysis, without the necessity of the implementation of such system to be completed.
In order to make an adequate performance assessment of Big Data-centered systems, though, we need a diversity of volumes and workloads that, due to the sensitivity of data concerning healthcare, might not be available. In other fields this problem is usually solved through the use of synthetic data generators, but in the field of healthcare these are few and not specialized for performance evaluation.
The main objective of this thesis is, thus, to create a synthetic data generator for the evaluation of the performance of a Big Data system model. The reference data for the generator will be gathered from the MIMIC-III dataset, a large, freely-available database comprising deidentified health-related observations (called "events") associated with over forty thousand patients who stayed in critical care units of the Beth Israel Deaconess Medical Center between 2001 and 2012 [3], and the MIMIC-III Waveform Matched Dataset, a dataset associated to MIMIC-III containing the digitalized vital signals (called "waveforms") recorded for the patients that stayed in the ICU wards of the hospital [4]. From now on, when talking about "MIMIC-III", unless clearly specified we will mean to consider both datasets together.
Since the architectural choices of the generator we intend to present are based on the information we are able to gather from MIMIC-III, we decided to split the body of our work in two main chapters: one dedicated to the analysis performed on MIMIC-III, and one dedicated to the software development of the generator itself.

2. Analisys of MIMIC-III

The objective of this analysis is to find a set of distributions to be used to model the time of acquisition of the events described in MIMIC-III.
To do so, we focused on the process of interaction between the patient and the system, specifically looking at the different events that are registered at each stage of such interaction process. Once these stages have been identified, we are able to look at the distribution of their duration and model it with some fitting techniques and tools. Finally, the same fitting procedures will be applied to model the inter-time between the registrations of the events contained in MIMIC-III.
Once the distributions of the duration of the interaction stages and of the inter-arrival time of the events for each stage will be available, we will be able to use them as the foundation for the synthetic data generator.

2.1. Design decisions for the analysis

Due to the sensitive nature of the data that comprises MIMIC-III, it had to undergo a de-identification procedure before the dataset could be made available to the public. In particular, for each individual patient the dates were shifted into the future by a random offset in a consistent manner. All the time intervals between two entries associated with the same patient are kept intact; other than that, only a handful of characteristics of the original date were still valid after the random shift, namely the day of the week, the general season, and the time of the day.
This procedure deeply influenced the structure of our work: an analysis of the exchange of data between the patients as a whole and the hospital system was in fact made basically impossible and, for this reason, we decided to focus on analysing the duration of the interactions of the patients singularly.

2.2. Stages of Interaction with the Hospital

After multiple refinements, the identified stages of the interaction process (represented as time intervals to be modeled) are:

1. The time passed in an ICU.
2. The time after the hospital admission and before the start of the first ICU stay.
3. The time in the hospital between two consecutive ICU stays.
4. The time between the end of the last ICU stay and the end of the entire hospital stay.
5. The total time in the hospital, which shall be considered as the sum of the times listed above (unless the patient is not admitted in an ICU, in which case the hospital time shall be computed separately).
6. The time between the end of a hospital stay and the beginning of the next one.

MIMIC-III considers two main categories of events: those associated with the specific ICU stay of the patient (like automatic measurements of their blood pressure made during their ICU permanence) and those associated with the general hospital stay of the patient (like laboratory results collected from their cultures). The former are generated only during the first stage of the ones listed above (the time passed in ICU), while the latter are generated through all the other intervals except the last one, during which no event shall be generated since the patient is not in the premise of the hospital. Each of these categories is then split into multiple kinds event, like laboratory events or note events, each stored in its own table in the dataset.

2.3. Classification

Before focusing on the distributions that can be fit to model the interaction stages identified in the previous section, we decide to split the events stored in MIMIC-III and the interactions found into "classes". This decision was taken to avoid considering a single distribution to model all the events produced by all the patients during their stays, which would risk worsening the fitting results.
The classes chosen are based on easily observable features of the data in order to maintain the grouping process simple. Such features, though, shall be relevant to the medical field and distinctive enough to split the dataset in comparable portions.
The features chosen for the classification are:

- the gender of the patient (Male or Female)
- the age of the patient (0-45, 45-65, 65-75, 75-100 and over 100 years old)
- the day of the week when the hospital stay begins.

These features lead to a total of 70 classes in which the patients and the admissions (and their events) can be split.

2.4. Distribution Fitting

After the classification has been performed, we are ready to fit some distributions to the interaction stages previously defined and to the events registered in MIMIC-III.
The duration of each interaction stage are fitted using Phase-Type distributions, commonly used in the performance evaluation world for their adaptability to any kind of empirical distribution [2], using the tool HyperStar. Since the tool requires the intervention of the user during the fitting process and we have 70 distributions to consider (one for each class) for each of the interactions identified, the time necessary to perform the fitting procedure would be excessive. For this reason, we opt to bring together the classes that have a similar empirical distribution. To find which these classes might be, we use the Kolmogorov-Smirnov two sample test, a test commonly used to assess whether two samples come from the same distribution [1].
For each of the interaction stages identified, we perform the test on every possible couple of classes, and group together those classes which have all the same distribution. In Figure 1 is shown a sample of the result, where the nodes represent the different classes, the arcs connect the classes with the same distribution according to the Kolmogorov-Smirnov test, and the circles represent the groups formed.

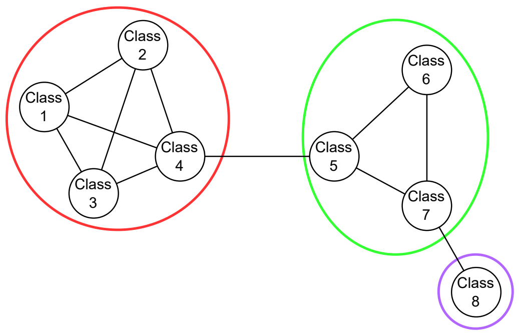

Figure 1: Grouping procedure applied on the classes

The events, on the other hand, are fitted using exponential distributions, one for each class and for each kind of event registered in MIMIC-III. The procedure chosen to fit an exponential to the inter-time between the recording of the events of each kind is based on the method of moments, which focuses on achieving the same average of the considered sample. Since most of the kinds of events registered in MIMIC-III have available one or more attributes that provide a clear indication of the time of their creation, this procedure is used for them. Those events that require a specific procedure to be fit due to the way they are structured are considered separately and specific procedures are employed for each of them.
The waveforms, too, are fitted using exponential distributions.

3. Development of the Generator

The objective features which we aim to obtain during the development of the generator are fine-tunability and adaptability.
Fine-tunability is intended as as the possibility of changing the parameters of the identified distributions from those obtained from the analysis performed and the ability to control the synthetic data generated as output.
Adaptability is intended as the ability to adapt the generator to different analysis procedures and, potentially, different datasets, without requiring excessive and complex modifications.
The generator is divided into 3 modules:

- **Configuration module**, which contains the necessary components for reading and managing the outputs of the analysis.
- **Classification module**, containing the components intended to model the classification made during the analysis phase.
- **Generation module**, containing the components needed to generate the synthetic events.

3.1. Configuration Module

In order to allow for an easier customization of the parameters of the generator, the management of the information obtained from the analysis has been centralized within a single module.
The configuration module consists in two main components:

- the **Manager class**, responsible for providing the other components of the generator with the information obtained from the analysis.
- the **configuration dictionary**, which con-

tains the default file paths where to find the outputs of the analysis.

To avoid the use of hardcoded strings in the other components of the generator, some enumerations are also introduced to enclose and group the keys of the previously mentioned dictionary.

As we will see later on, these are also used to model the interaction stages identified during the analysis and the kinds of events associated with each of them.

3.2. Classification Module

The classification module collects the enumerations intended to model the groups obtained by the classification procedures previously described. To do so, it uses two enumerations, namely `PatientClass` (used to model the classifications based on the gender and the age of the patients) and `AdmissionClass` (which models the classifications based on the weekday at which the hospital stay started).

3.3. Generation Module

The generation module is the main module of the generator, in charge of the generation of the synthetic events from the distributions fitted during the analysis. Its components follows a layered structure, which is shown in Figure 2. Each layer is comprised of a single class that implements the `EventsGenerator` interface, which defines the `get_events` and `get_waveforms` methods, used to retrieve the events and the waveforms that result from the generation.

Each component, when asked to generate the events, creates an instance of the component of the following layer and routes the request to them, up until the `Interaction` layer is reached, which ultimately generates the events.

This structure was chosen to allow the user to finely control the events to be generated by choosing which layer to request the generation of the events to. For example, if the user is interested in the generation of the events of multiple patients, regardless of their classification group, he can simply use the `Hospital` class, which once provided with the number of patients to consider and the time distance between their first admission to the hospital is able to generate the required events. If instead the user would like to consider the events of the patients of a specific age group, it could use the `Patient` class and specify it for each of the instances.

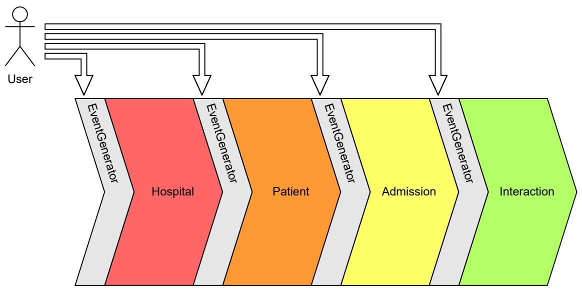

Figure 2: The layered structure adopted for the generation module.

The classification group to consider is decided by the Hospital layer (which chooses the age and the gender of the patient) and by the Patient layer (which chooses the day of the week on which the admission begins). The chosen classification group is passed on to the following layers by the corresponding element of the enumeration discussed in Section 3.2.

The distributions to be used for the generation of the events and the other outputs of the analysis (like the probability of the patient having a certain age or a certain gender) are requested by the components of the generation module to the `Manager` class. The `Manager` class is meant to be instantiated by the user and provided with the configuration dictionary (the default configuration dictionary previously introduced in the configuration module is used otherwise) to retrieve the outputs of the analysis; the instance created can then be provided to the layer the user intends to use. During the creation of the following layers, the instance of the Manager will be passed on to allow the generation of the events.

As previously stated, the only layer that effectively performs the generation of the events is the **Interaction** layer. This layer models the stages of the Interaction process identified previously and, differently than the other layers, contains more than one class. In fact, the Interaction layer contains two classes, both extending the abstract class *Interaction*:

- the `StayInteraction` class, which models all the interaction stages that generate the events associated with the entire hospital stay.
- the `ICUInteraction` class, which models

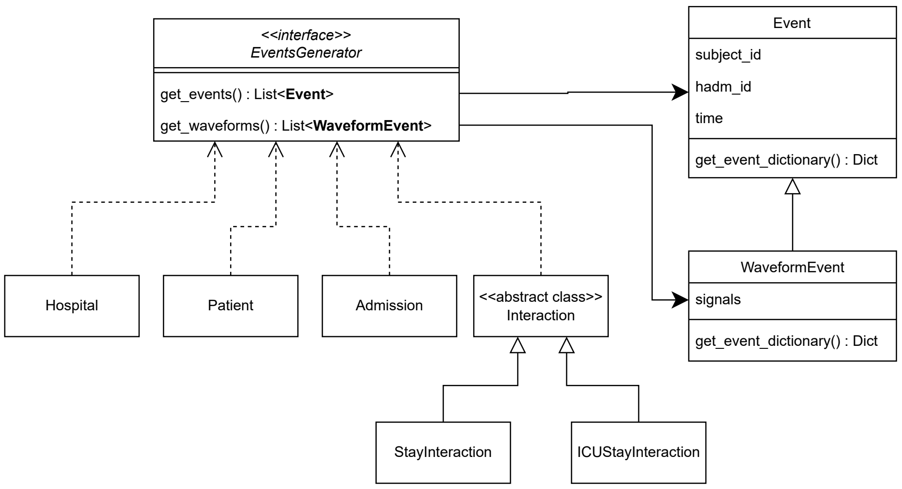

Figure 3: UML diagram of the Generation Module.

all the interaction stages that generate the events associated with only the ICU stay.
To generate the synthetic events, the two classes request to the instance of the `Manager` class (either passed on by the previous layers or provided by the user) the parameters of the exponential distributions fitted during the analysis. The duration of the interaction is provided by the Admission layer, which requests the necessary parameters to the Manager to generate it according to the phase-type distributions fitted during the analysis; during this exchange, information about the specific interaction stage considered is communicated through the elements of the enumerations introduced in the configuration module.
The events are represented in the generator by the `Event` class, regardless of their category or type. The waveforms are represented by an extension of such class.
A high level UML diagram of the generation module is shown in Figure 3.

4. Experiments and Results

In order to show the procedure to follow to use the generator presented in the previous chapter and to evaluate its characteristics, we consider two illustrative experiments based on two fictitious scenarios.
Both scenarios involve the evaluation of the performances of a Big Data architecture intended to be used as the main component of a hospital's information system that receives and processes the same events stored in MIMIC-III.

In the **first scenario**, the system is to be used for managing all the data covered by the generator. In the **second scenario**, the system is to be used as the hospital's ICU data system, and provisionally it is intended to be used to manage only the structured data generated during the patients' ICU stays.
In both scenarios, our goal is to evaluate the possibilities of downgrading or upgrading the considered system by assessing its performances using the synthetic data generator described in this paper.
The performance analysis and the design of the model is performed using *JMT*, "a suite of tools for the performance evaluation of computer systems" [5].
In particular, for the first scenario we need the events and the waveforms to be generated for multiple patients, without any specific requirement. For this purpose, we used the Hospital layer of the generator.
For the second scenario, on the other hand, we need to only generate the events associated with the ICU stays. For this reason, we used the Interaction layer of the generator, more specifically the `ICUInteraction` class.
With the generated events, we are able to evaluate the system's performances, showing through an analysis of various performance indices (such as throughput, response time, and utilization) that, considering a limited increase in its service time, the performances of the system do not change drastically. It follows from these considerations that a downgrade (albeit limited) of the system under consideration is possible.

While evaluating the second scenario, with the aim of showing the adaptability of the generator, we chose to simulate a change of the chosen classification features by modifying the outputs of the analysis performed on MIMIC-III; in particular, the feature changed was the age of the patient, which was modified to consider different age ranges. This change required only a minor modification to the classes of the generator; in fact, it was sufficient to modify the enumerations described in the classification module.
Through the experiment conducted on the two scenarios, we are able to show two important qualities of the newly created synthetic generator: the fact that it can be adapted to multiple use cases and customized easily, in line with the goal of *adaptability* previously identified, and the fact that the generated output can be easily controlled by the user, in line with the goal of *fine-tunability*.

5. Conclusions

In this work, we have developed a synthetic data generator to be used for the evaluation through modeling techniques of a Big Data System.
As a reference for the creation of the generator, we relied on MIMIC-III, a publicly available dataset containing the measurements and observations made on patients of a hospital during a multiple years period and recorded by the data system of the hospital in question.
That dataset has a new version available, called MIMIC-IV, which has a different structure and considers a longer time frame. To date, it does not yet contain the waveforms records, but their introduction has been initiated. A possible future work might involve using this new dataset as a reference for a new version of the generator.
With the aim of obtaining additional information on the temporal distribution of such events, we first analyzed the interaction process between the patients and the hospital system, focusing on the time periods during which the events described in MIMIC-III were recorded in the system. These time periods were then modeled through appropriate distributions, taking advantage of distribution fitting techniques and tools; these same techniques were then used to model the inter-times of each kind of event.
The results obtained, such as the parameters of the fitted distributions and the structure identified for the interaction process between the patients and the hospital system, were used as the foundation for the generator's architectural choices.
To allow a granular control over the output of the generator, a layered architecture was adopted during its development. Moreover, throughout development, the ability to customize the generator was foregrounded, to make it easy to adapt it to different analysis procedures.
Finally, to test the possibilities just described, two experiments were carried out, focused on evaluating the performance of a simple model of a Big Data system in two different scenarios. They resulted in a success, showing how easy it is to tune the output of the generator and to change its components.

References

[1] Vance W. Berger and YanYan Zhou. Kolmogorov–Smirnov Tests. In *Encyclopedia of Statistics in Behavioral Science*. John Wiley & Sons, Ltd, 2005.

[2] Mogens Bladt. A Review on Phase-type Distributions and their Use in Risk Theory. *ASTIN Bulletin: The Journal of the IAA*, 35(1):145–161, May 2005.

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

[4] Benjamin Moody, George Moody, Mauricio Villarroel, Gari Clifford, and Ikaro Silva. MIMIC-III Waveform Database Matched Subset. 2017.

[5] G. Serazzi, G. Casale, and M. Bertoli. Java Modelling Tools: an Open Source Suite for Queueing Network Modelling and Workload Analysis. In *Third International Conference on the Quantitative Evaluation of Systems - (QEST'06)*, pages 119–120, September 2006.