Visual Analytics for Medical Workflow Optimization

Visual Analytics for Medical Workflow Optimization

Humberto Simón García Caballero

Visual Analytics for Medical Workflow Optimization

Humberto Simón García Caballero

Visual Analytics for Medical Workflow Optimization

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van
de rector magnificus prof. dr. ir. F.P.T. Baaijens,
voor een commissie aangewezen door het College
voor Promoties in het openbaar te verdedigen op
vrijdag 12 november 2021 om 13:30 uur

door

Humberto Simón García Caballero

geboren te Cieza, Murcia, Spanje.

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

voorzitter:	prof. dr. J.J. Lukkien
1^{e} promotor:	prof. dr. M.A. Westenberg †
2^{e} promotor:	prof. dr. ir. J.J. van Wijk (prof. dr. ir. J.J. van Wijk vervangt wijlen prof. dr. M.A. Westenberg als eerste promotor)
leden:	prof. dr. B. Preim (Otto-von-Guericke Universität Magdeburg) prof. dr. J. Kohlhammer (Technische Universität Darmstadt) prof. dr. ir. B.F. van Dongen prof. dr. J.B.T.M. Roerdink (Rijksuniversiteit Groningen) prof. dr. S. Overeem prof. dr. A. Vilanova

Het onderzoek of ontwerp dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

Colophon

The work in this dissertation is part of the Data Science Flagship program, which is financed by Philips Electronics Netherlands

Printed by: Gildeprint - Enschede

Cover design: Humberto Simón García Caballero

Cover background: Solen Feyissa on Unsplash

A catalogue record is available from the Eindhoven University of Technology Library.
ISBN: 978-90-386-5395-2

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

An electronic version of this dissertation is available at http://repository.tue.nl

Sábete, Sancho, que no es un hombre más que otro si no hace más que otro:
todas estas borrascas que nos suceden son señales de que
presto ha de serenar el tiempo, y han de sucedernos bien las cosas,
porque no es posible que el mal ni el bien sean durables,
y de aquí se sigue que, habiendo durado mucho el mal,
el bien está ya cerca.

Don Quijote de la Mancha — Miguel de Cervantes Saavedra

¿Qué es la vida? Una ficción,
una sombra, una ilusión,
y el mayor bien es pequeño;
que toda la vida es sueño,
y los sueños, sueños son.

La vida es sueño — Pedro Calderón de la Barca

Contents

Colophon **v**

1 Introduction **1**
1.1 Motivation . . . 2
1.2 Objective . . . 3
1.3 Outline & Contributions . . . 4
1.4 Publications . . . 6

2 Background **9**
2.1 Introduction . . . 10
2.2 Medical Workflows . . . 10
2.2.1 Users, Tasks and Problems . . . 10
2.3 Workflow Analytics: Optimizing Medical Workflows . . . 11
2.3.1 Process-Centric Perspective . . . 11
2.3.2 Machine Learning Perspective . . . 13
2.3.3 Sleep Staging . . . 14
2.3.4 Machine Learning in Sleep Staging . . . 16
2.4 Visualization . . . 17
2.5 Visual Analytics . . . 19
2.6 Visualization in Process Mining . . . 20
2.7 Visualization in Sleep Staging . . . 21
2.8 Conclusions . . . 22

3 Soundness Analysis in Petri nets **25**
3.1 Introduction . . . 26
3.2 Problem Definition . . . 27
3.2.1 Problem Analysis . . . 27
3.2.2 Soundness Validation . . . 27
3.3 PSVis . . . 28
3.3.1 Glyphs on the Petri net . . . 29
3.3.2 Petri Net View . . . 29
3.3.3 States View . . . 30
3.3.4 Runs View . . . 33
3.3.5 Design Decisions . . . 33
3.3.6 Implementation Details . . . 34
3.4 Use Cases . . . 34
3.5 Conclusion . . . 35

4 Performance and Conformance Checking for Process Models **37**
4.1 Introduction . . . 38

4.2 Related work . . . 39
4.2.1 Sequence alignment-based techniques . . . 40
4.2.2 Process analytic based techniques . . . 40
4.3 Dataset and Tasks. . . . 41
4.4 Design Decisions . . . 42
4.4.1 Icon design . . . 43
4.4.2 Activity links design . . . 44
4.5 SepVis . . . 44
4.5.1 Pathways View . . . 46
4.5.2 Activity filters and conformance checking . . . 48
4.5.3 Temporal filters: Performances . . . 49
4.5.4 Attribute filters: Cohort distributions . . . 49
4.5.5 Implementation . . . 50
4.6 Use Cases and Results . . . 51
4.6.1 Clinical pathways in NC and IC units . . . 51
4.6.2 Performances and delays . . . 51
4.6.3 Deviations and clinical attributes . . . 52
4.7 Conclusion . . . 53

5 Interactive Correction of Deep Learning Predictions in Sleep Staging 55
5.1 Introduction . . . 56
5.2 Medical Background . . . 57
5.3 Related Work . . . 58
5.3.1 Performance Analysis. . . . 59
5.3.2 Neural Network Analysis . . . 59
5.3.3 Model Interpretation . . . 59
5.3.4 Explanation Techniques . . . 60
5.4 Problem Definition . . . 60
5.4.1 Model Description and Dataset . . . 61
5.4.2 Tasks . . . 63
5.5 V-Awake . . . 63
5.5.1 Cohort View . . . 65
5.5.2 Predictions View . . . 65
5.5.3 Dimensionality Reduction View . . . 67
5.5.4 Selections View. . . . 68
5.5.5 Input View . . . 69
5.6 Use Case . . . 70
5.6.1 Exploration Patient 1 . . . 70
5.6.2 Exploration Patient 2 . . . 72
5.7 Discussion and Limitations . . . 73
5.7.1 Approach Generalization . . . 74
5.7.2 Scalability . . . 74
5.8 Conclusions . . . 75

6 Explainability for Sleep Staging 77
6.1 Introduction . . . 78

6.2 Images and Time Series . . . 79
6.3 Case Study . . . 80
6.3.1 Model . . . 80
6.3.2 Dataset . . . 81
6.3.3 Model Explanation . . . 82
6.3.4 Does the model predict the same by occluding the input? . . . 83
6.3.5 Learned Filters . . . 85
6.4 Discussion . . . 87
6.5 Conclusions . . . 88

7 Performance Assessment of Sleep Staging Models 89
7.1 Introduction . . . 90
7.2 Medical Background . . . 91
7.3 Problem Definition . . . 92
7.4 Related Work . . . 94
7.5 PerSleep . . . 95
7.5.1 Model Selection . . . 95
7.5.2 Patient Data . . . 97
7.5.3 Performance View . . . 99
7.5.4 Physiological Data View . . . 101
7.5.5 Complexity and Scalability . . . 102
7.6 Use Case . . . 103
7.6.1 Sleep Fragmentation . . . 106
7.7 Discussion . . . 106
7.7.1 Approach Generalization . . . 107
7.8 Conclusions . . . 108

8 Conclusions 109
8.1 Achievements . . . 110
8.2 Research Question . . . 111
8.3 Directions for Future Research . . . 114
8.4 Lessons Learned . . . 115
8.5 Finally . . . 116

References 117

Summary 133

Curriculum Vitæ 135

Acknowledgments 137

1

Introduction

This chapter introduces the topics of process mining and machine learning and their influence in healthcare. We next present our research question and provide an overview of the work discussed in this dissertation.

1

1.1 Motivation

Nowadays, processes and process data are everywhere. From purchasing a product from a webshop to receiving a treatment at a hospital, many daily life activities can be modeled as processes. Formalization of processes is useful to support analysis, performance assessment and improvement, among others. Many of these benefits are possible due to advancements in two areas: process mining and machine learning. Process mining is a relatively young field that can be positioned between machine learning and data mining on one side, and process modeling and analysis on another [170]. Through knowledge extraction from event logs, process mining aims to discover, monitor and improve processes (i.e., workflows). The performance of processes can also be improved by focusing on a specific task in a workflow. For instance, a manual task could be replaced by an automated one. Machine learning is defined as the study of computer algorithms that improve automatically through experience [108]. Experience is gained through an iterative process in which properties of the input data are exploited, thereby enabling automating tasks in workflows.

Medical workflows are of special interest because they are ubiquitous, often complex, and, above all, errors can have a huge impact on the welfare of persons. Typical examples of medical workflows are guidelines to treat diseases, performing laboratory tests and admission or discharge procedures. Ensuring the correct execution of such workflows in a timely manner is crucial to deliver proper healthcare. The analysis of workflows can provide valuable insights into how current behavior compares to ideal behavior within a hospital and can help to find design flaws of workflows. Furthermore, the automation of certain parts of the workflow can speed up the whole process considerably.

Process mining has proven to be effective in healthcare environments for different purposes [137]: process discovery [17, 102, 103], conformance checking [82, 201], and social network analysis [17, 89, 102]. Conformance checking is useful to provide healthcare experts with knowledge on how to adjust to internal and external guidelines; performance analysis can help to identify and resolve possible bottlenecks to reduce waiting times.

Many advancements in the history of healthcare have improved the quality and prospect of life. From developments in vaccines to progress in technology, they have led to more preventive, analytical healthcare that focuses on patient-centered care. From the technology perspective, data analysis, and in particular machine learning, is attracting much interest and represents a promise for the future of healthcare. Examples that currently apply machine learning in healthcare can be found in academia [22, 69] and industry [11, 42, 94, 132]. In general, machine learning can be used to help doctors providing more personalized diagnostics and treatments, and reduce the costs for such care treatment [14].

Progress in machine learning promises the incorporation of automated techniques in medical workflows. Areas ranging from image analysis to multivariate data analysis can benefit. Also, clinicians often have to deal with temporal data. For example, in sleep analysis, the sleep of a patient is recorded during the night, scored by a technician and then analyzed by a somnologist to diagnose possible sleep disorders. Much progress has been made in automatizing the scoring step where automated techniques would be used to score the sleep of the patient. Most techniques employ the same data that has been traditionally used in the

manual approach: electroencephalography (EEG). Moreover, in recent years, new sources of data such as surrogate devices (e.g., wrist-worn trackers) have changed the way sleep is measured and so the models utilized. These new data sources entail the ability to capture different physiological aspects. However, the precision of such methods is usually not comparable to the gold standard. This variety of data sources and models has resulted in a strong need to evaluate and improve the performance of such models to ensure their reliability for real-world usage.

1.2 **Objective**

The research question presented in this dissertation is twofold: it concerns both global workflow analysis and particular task improvement. The main question can be formulated as follows:

How can we use interactive visualization and automated techniques to understand and optimize medical workflows?

To tackle this research question, we focus on global workflow analysis (process mining) and local task improvement (machine learning). Because these fields are too broad, we focus on four topics, namely soundness verification, conformance checking, machine learning to support time series analysis and for different groups of patients. Within the process mining perspective, soundness verification and conformance checking are very important. First, soundness verification ensures that the modeled or discovered process is sound. Once the process fulfills this criterion, it can be implemented in, for instance, a hospital setting. Once the process is being executed, certain deviations can occur. To this end, conformance checking can help to analyze these deviations and provide a better understanding of how the process is executed in reality in a hospital setting. Regarding the machine learning perspective, we focus on the sleep disorder diagnosis. In particular, machine learning can be useful to automatize certain parts of the diagnosis process. For example, sleep staging can be done automatically with machine learning models. In this context, gaining more knowledge on the model utilized is crucial. This can be done in two different ways. One way is by analyzing the inner workings of the model to understand what features are utilized to make a prediction. Another way is by evaluating the predictive performance of such a model. In order to address these topics, we aim to answer the following questions throughout this dissertation.

1. *How can we understand the circumstances under which soundness breaks down in a Petri net?* Soundness is a desirable property in process models. It depicts a notion of correctness. During the designing of a process model, soundness has to be ensured such that certain unwanted situations are avoided. For example, a typical soundness check is to verify that the process model contains no dead transitions (i.e., transitions that cannot be executed). Traditionally, soundness verification has always been done automatically. This produces a single answer stating whether or not a process model was sound or not, which is too limited to act upon. The challenge, therefore, lies in understanding under what circumstances soundness is violated. Models can get very complex not only in their graphical, static representation but also in their behavior

(i.e., semantics). Combining automated techniques from process mining with visualization techniques can help to tackle these challenges.

2. *How can we gain insights into the modeled and real processes of hospitals with conformance checking?* Generally, hospitals use guidelines to treat certain diseases. These guidelines include drug administration, monitoring, test running, etc. Process mining provides automated techniques that can be used to discover process models and assess the conformance with real behavior. The latter can indicate whether the personnel in the hospital matches the guides suggested in the guideline. Therefore, conformance assessment is crucial to understand if certain diseases are treated according to the guidelines. The problem with this sort of automated analysis is that it produces a large amount of data that is rather difficult to analyze. Moreover, patient data (e.g., other conditions, laboratory results) is not taken into account. Nevertheless, patient data can provide more insights into the decisions taken by doctors to provide certain drugs or diagnoses, which can justify deviations from the guidelines. A visual exploration of this process data in combination with patient data can alleviate these limitations and provide the users with more detailed insights into the processes running at hospitals, and how they relate to patient data.

3. *How can we support the usage of machine learning in a real-world, clinical setting?* When a machine learning model has been fully developed and is ready to be used in a real-world setting, the users of these models are left to simply trust the output of the model to a certain degree. Such trust is based on two factors: previous knowledge of the model (e.g., performance evaluation) and the probabilities produced by the model. However, often models are not perfect and can struggle to classify difficult cases and yet output relatively high confidence values. In medical settings, it is utterly necessary to ensure the correctness of the outputs of the model, especially if diagnoses are influenced by these outputs. Therefore, the challenge resides in enabling an efficient exploratory process through a sheer amount of data to find potential misclassifications such that experts can ensure the correctness of the outputs produced by machine learning models without losing the benefits of automation.

4. *How can we evaluate the performance of a machine learning model for different groups of patients?* The performances of machine learning models vary greatly depending on the problem addressed, the data used during training and the architecture of the model itself. In the medical domain, these models can be applied to patients with underlying physiological differences such as age, sex, medical conditions, etc. This makes the analysis of the performance much more complex, as the variation in patients has to be taken into account to make proper assessments. Visual analytics can be the means to address this problem.

1.3 **Outline & Contributions**

This thesis addresses the needs for analyzing and improving workflows in the medical domain. This is done by looking at the research question from two perspectives, namely process-centric and machine learning approaches (see Figure 1.1). In the process-centric approaches,

the aim is to analyze medical workflows to gain insights. That can be used to refine workflows to guarantee certain desirable properties. In addition, process-centric approaches can be used to verify whether daily work in a hospital conforms to or deviates from a guideline. Machine learning approaches are used to automate certain tasks in the medical domain. This thesis focuses on sleep staging, where classifiers are used to score the sleep of persons. Sleep staging is chosen because of the industrial partners involved in this project. Nevertheless, our research can be generalized to other domains that share features with sleep staging (e.g., sequential data, health-related and physiological signals).

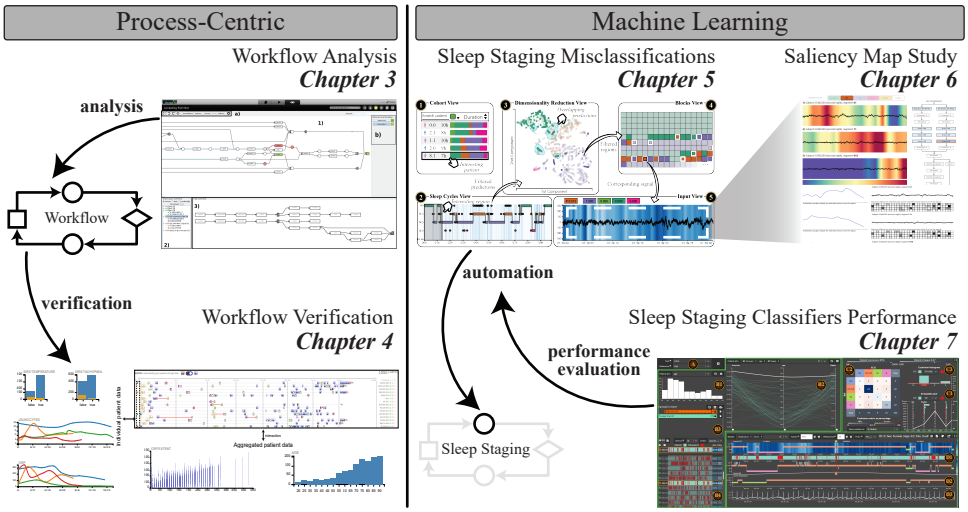


Figure 1.1: Overview of the structure of this dissertation. It is divided into two main categories: process-centric and machine learning approaches.

Research areas that are relevant to our research question include visualization, process mining, machine learning, healthcare and sleep staging. Chapter 2 presents an overview of these fields and related work.

In the first part of this dissertation, we present process-centric techniques. Soundness is a desirable property in process models. Determining whether a process model is sound can be done automatically with computation. However, understanding why a model is not sound is a challenging task that involves complex exploration. In chapter 3 we present a novel method to explore the soundness of process models in the form of Petri nets.

Deviations from guidelines occurring in hospitals are important to verify flaws in the design of workflows. Conformance checking is an automatic method to compute such deviations. Although this automated technique is informative, it remains difficult to explore the deviating cases. To this end, chapter 4 presents a method that combines process mining techniques and visualization. It enables the exploration of the outputs of conformance checking algorithms to verify whether guidelines and daily work match.

The second part of this dissertation focuses on machine learning techniques to improve specific tasks of workflows. In particular, we focus on sleep staging, which is the process of

scoring the sleep stages of patients during their sleep. This task can be automated utilizing machine learning models. However, this automation also brings new challenges. In chapter 5 we present the first visual analytics system to help to find misclassifications in a ground-truth free scenario in sleep staging. We also discuss the generalization of our approach to other domains.

The approach presented in chapter 5 utilizes saliency maps. These are used to visualize the parts of the input data that are important for the model to make a prediction. We study the effect of different perturbations in the input data for the generation of saliency maps in chapter 6.

When a classifier is used in a real-life scenario, especially in the medical domain, it is important to ensure that its users understand how the model behaves. This is especially important in sleep staging, where the temporal nature of the data poses new challenges that are difficult to address with traditional approaches. To this end, we present a visual analytics system in chapter 7 to assess the performance of sleep staging classifiers.

In chapter 8 we summarize the major findings from chapters 3 to 7 and answer the research question based on these findings. In addition, we provide directions for future work.

1.4 **Publications**

Publications in scientific conference proceedings and journals:

- **Garcia Caballero, H. S., Westenberg, M. A., Verbeek, H. M. W., and van der Aalst, W. M. P.** Visual analytics for soundness verification of process models. In *Business Process Management Workshops* (Cham, 2018), E. Teniente and M. Weidlich, Eds., Springer International Publishing, pp. 744–756 [53].
  This publication serves as core material for chapter 3.
- **Garcia Caballero, H. S., Corvò, A., Dixit, P. M., and Westenberg, M. A.** Visual analytics for evaluating clinical pathways. In *2017 IEEE Workshop on Visual Analytics in Healthcare (VAHC)* (2017), pp. 39–46 [50].
  This publication serves as core material for chapter 4.
- **Garcia Caballero, H. S., Westenberg, M. A., Gebre, B., and van Wijk, J. J.** V-awake: A visual analytics approach for correcting sleep predictions from deep learning models. vol. 38, pp. 1–12 [52]. (**Honorable Mention Award**).
  This publication serves as core material for chapter 5.
- **Garcia Caballero, H. S., Westenberg, M. A., and Gebre, B.** Explainability for one dimensional temporal inputs of deep learning models. *Demo at the 1st Workshop on Visualization for AI explainability (VISxAI)* (2018). Online publication [51].
  This publication serves as core material for chapter 6.
- **Garcia Caballero, H. S., Corvo, A., van Meulen, F., Fonseca, P., Overeem, S., van Wijk, J. J., and Westenberg, M. A.** Persleep: A visual analytics approach for performance assessment of sleep staging models. In *Eurographics Workshop on Visual Computing for Biology and Medicine, VCBM 2021* (2021), The Eurographics Association [49]. (**Honorable Mention Award**).
  This publication serves as core material for chapter 7.

Publications to which I contributed during my PhD but that are not included in the dissertation:

- **Dixit, P. M., Garcia Caballero, H. S., A., C., Hompes, B. F. A., Buijs, J. C. A. M., and van der Aalst, W.** Enabling interactive process analysis with process mining and visual analytics. In *Proceedings of the 10th International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 5: ACP, (BIOSTEC 2017)* (2017), INSTICC, SciTePress, pp. 573–584 [39].
- **Corvò, A., Garcia Caballero, H. S., and Westenberg, M. A.** Survivis: Visual analytics for interactive survival analysis. In *10th International EuroVis Workshop on Visual Analytics, EuroVA@EuroVis 2019, June 3, 2019, Porto, Portugal* (2019), Eurographics Association, pp. 73–77 [30].
- **Corvò, A., Garcia Caballero, H. S., Westenberg, M. A., van Driel, M. A., and van Wijk, J.** Visual analytics for hypothesis-driven exploration in computational pathology. *IEEE Transactions on Visualization and Computer Graphics* (2020), 1–1. Early access [31].

2

Background

In this chapter, background information is provided on the concepts discussed in this dissertation. We start by describing medical workflows and the main users, tasks and problems that we consider in this dissertation. We then elaborate on the topic of optimizing medical workflows and describe process mining, machine learning and sleep staging. We next introduce the fields of visualization and visual analytics. Finally, we discuss the role of visualization in process mining and sleep staging.

2.1 Introduction

This dissertation addresses the analysis of medical workflows from two different perspectives: process-centric and machine learning. To provide the reader the necessary background information, we first introduce the topic of optimizing medical workflows where process-centric and machine learning perspectives are discussed. Next, we introduce data visualization and visual analytics and present the role of visual analytics in these disciplines. Further details, as well as related work about specific background concepts, are provided in the corresponding background sections in the following chapters of this dissertation.

2.2 Medical Workflows

Workflows are made of several tasks that are usually executed by people and involve resources. The ultimate aim of a workflow is to produce an output or to achieve a goal within an organization, for instance, a company, hospital or department. Medical workflows, in particular, focus on delivering a certain benefit to patients: treatments, diagnoses, laboratory tests, etc. They target the welfare of patients and pursue their recovery. For this reason, workflows and the tasks they involve play a fundamental role in healthcare.

2.2.1 Users, Tasks and Problems

The analysis of workflows enables experts to investigate design flaws and optimize certain tasks that are performed within workflows. In this section, we provide some examples of experts that can benefit from interactive visualizations in those domains.

Process mining expert
Processes must be reliable and predictable. Although the definition of a process is static, the behavior it can generate is certainly not. When process mining experts design or mine process models, they want to ensure certain properties. Due to the dynamics in the behavior of the model, it is challenging to explore and assess these properties in a traditional manner.

Healthcare workflow analyst
Workflows running in a healthcare setting are dynamic, complex and multidisciplinary. Improving workflows in healthcare may impact the quality of life of patients [137]. One possibility of improvement is conformance checking. Assessing how process models match real behavior can provide valuable insights to analysts. This can be useful, for instance, to verify how hospitals adhere to internal and external guidelines and protocols, find deviating behaviors and contextualize them. Visualization can facilitate the exploration of the conformance checking to be more effective and efficient.

Sleep staging technician
Sleep staging can benefit from automated techniques to speed up the whole process. This automation requires supervision from technicians to ensure the correctness of the results of

sleep staging. If the entire output space needs to be verified, the benefit introduced by automation vanishes. However, enabling an efficient exploration of the data in a ground-truth free context is a challenging problem. Visualization can alleviate this exploration through coordinated views and user interaction.

Machine Learning Expert in Sleep Staging
Machine learning experts need to validate the models they create for sleep staging. They usually rely on traditional metrics like predictive accuracy or kappa value to assess how the model performs overall. Sleep staging, however, poses certain challenges that cannot be captured with individual metrics. For instance, sleep fragmentation can happen at any time of the night. Thus, calculating a global metric depicting the sleep fragmentation would not be sufficient to determine whether a model is recognizing this characteristic correctly at different moments of the night. Moreover, models in sleep staging are applied to a wide variety of patients with different characteristics. Visualization can mediate in the assessment process to get insights into the ML model, for instance, to check if the model responds equally well for different types of patients.

2.3 Workflow Analytics: Optimizing Medical Workflows

The analysis of workflows can become difficult due to the sheer size of tasks, and a large number of connections between these. For this reason, interactive visualizations may not suffice to provide insights. Supporting the analysis with automated techniques can help users to make sense out of their workflow models.

Optimizing medical workflows can be achieved in two different ways: improving the workflow as a whole, and optimizing some key tasks. In this section, we will address these two different paths by explaining process-centric techniques that can be used to improve the workflow as a whole, and machine learning approaches to optimize specific tasks.

2.3.1 Process-Centric Perspective

In the last decades, there has been a significant shift from *data-centered* to *process-centered* systems [173]. In this context, process mining has gained interest regarding information systems. In these systems, information is extracted from event logs to perform process monitoring and improvement [174]. Such event logs are usually generated by information systems such as Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) systems in particular, and any transactional database that stores events in general. The collection of methods and algorithms that can mine and analyze workflows out of event logs is known as process mining.

Workflows can be defined in different formats, being token-based semantics one of the most common notations [174]. In this regard, Petri nets [125] have gained much traction because of their formal definition, which enables formal proof of desirable properties; the ability to model concurrency, which is a fundamental part of processes; and the availability of many

analysis techniques, which allow experts to, for instance, align modeled and observed behavior. In general, a Petri net can be defined as a bipartite graph composed of *places* and *transitions*. Although its structure (syntax) is static, behavior (semantics) can be dynamic. The semantics of a Petri net can be described with tokens and the way they flow through the model. In general, places contain zero or more tokens, and transitions consume one or more tokens. These represent the building blocks of Petri nets.

Certain properties are desirable in Petri nets. For an organization, it is vital to ensure the correctness, effectiveness, and efficiency of a business process. In particular, *soundness* [2], which is a notion of correctness, is a desirable property when designing a workflow using Petri nets. When a Petri net is sound, the following criteria are met:

- Starting from the initial place, it is always possible to reach a final state with only one token in the places that represent final states;
- when a final state is reached, the other places are empty; and
- the Petri net contains no dead transitions, i.e., transitions that cannot be executed at a certain state.

The first requirement indicates that a final state is always reachable, which prevents livelocks. Sadiq and Orlowska [140] were among the first in stating that modeling workflows could lead to deadlocks and livelocks. Essentially, livelocks can occur when the semantics of a model allow for behavior that can end up in an infinite *loop*. The second requirement refers to the so-called *proper termination* [57], which is considered a desirable property for models to properly operate. Finally, the last criterion avoids cases in which a certain transition cannot be executed.

In a healthcare setting, process models are used to model the desired flow of work and the medical protocols [98]. Figure 2.1 shows a simple example of a process model as a Petri net. The guidelines and protocols from a hospital can be modeled using such process models. It should be noted that alongside sequences, most of the process modeling notations allow for rich behavioral aspects, such as choice in between multiple activities, concurrent execution of activities, repetition of a particular fragment in a model, etc.

When a workflow is deployed for usage in a certain institution, it is desirable to assess how well it fits the actual work done in such an institution. Conformance analysis [172] essentially projects an event log onto a process model to help find deviating behaviors, that is, workflow paths that do not align with the process model. More specifically, conformance analysis finds a *path* from start to end in the process model that best describes a case from the event log by using a cost function. It also provides insights into how well an event log fits a particular process model, and how and where the process deviates from the protocols/guidelines in place. Next to this, alignments can also be used to investigate the possible bottlenecks in the process. For each event in the event log, the following three possibilities can arise (see *possible alignments* in Figure 2.1):

Synchronous move Occurrence of an event in a trace that is allowed by the current state of the model. The state of the process is given by the execution of previous events in a trace.

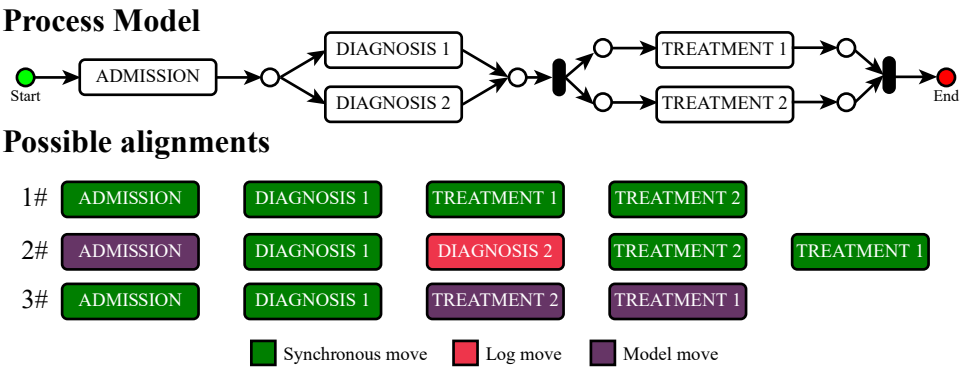


Figure 2.1: A simple process model in a hospital setting and a set of possible alignments. The first step in the process is Admission, followed by either Diagnosis 1, or Diagnosis 2. Diagnosis is followed by both Treatment 1 and Treatment 2, which can occur in any order.

Log move Occurrence of an event in the trace that is not allowed by the current state of the process. The second trace in Figure 2.1 contains a log move because *diagnosis 2* happens in the trace although it is not allowed by the model (only *diagnosis 1* or *diagnosis 2* can happen, but not both.

Model move The current state of the process cannot continue with the event that is being treated in the trace. The second trace in Figure 2.1 starts with a model move since the model requires to begin with *admission* activity. However, this activity is missing in that trace. Similarly, the third trace contains two model moves at the end, because both *treatment 1* and *treatment 2* are mandatory (notice that they can happen in any order).

2.3.2 Machine Learning Perspective

Machine Learning (ML) has gained much traction in the last decade due to huge advancements in the field. Essentially, ML is about defining algorithms that can learn new concepts, models or abstractions. Through a *training* process, the algorithm builds a model based on patterns discovered within the data. When the training process is finished, the built model can be used to predict or classify new data instances.

One technique that has remarkably progressed in the last years are neural networks, and especially Deep Learning [143] (DL). This technique tries to mimic the behavior of the human brain where interconnected neurons react to input signals to produce an output. Similarly, neural networks use layers of neurons that interact with the input data to generate an output. Research by Rumelhart et al. [138] revealed how large numbers of layers can be trained, which led to the success of DL nowadays. The output of these models is usually in the form of a vector of probabilities, where each probability indicates the likelihood of the input data of being of a certain class. In most cases, the maximum probability is taken as the final output of the model. Figure 2.2 shows a depiction of a neural network.

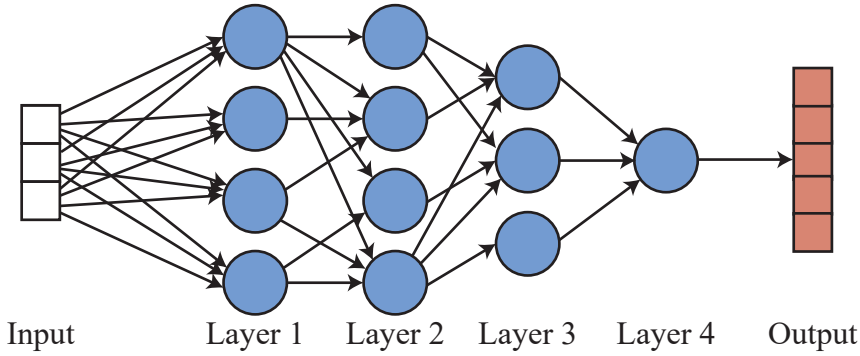


Figure 2.2: Depiction of a neural network. Each layer contains a set of nodes that are connected to nodes in the previous and next layers. The first layer represents the input data, while the last layer produces the output, which is usually a class.

ML techniques can speed up certain tasks, improving medical workflows. For instance, it may be used to help clinicians in detecting certain objects in radiology scans, tomography images or colonoscopy videos [5, 33, 68, 150, 169].

2.3.3 **Sleep Staging**

Sleep is a natural process common to a broad variety of species [20] such as mammals, birds, reptiles, insects, etc. Although it is an ancient process linked to our daily behavior, it was not until the 18th century when scientists started to wonder about sleep and its effects on health. One of the first scientists who studied the periodicity of sleep in plants was Jean-Jacques d'Ortous de Mairan. In his study from 1729, he demonstrated the existence of circadian rhythms in plants by looking into the folding and spreading of the leaves [202]. This raised interest in sleep behavior in humans. In 1912 Henri Piéron [129] published one of the first books that treated the health-related issues of sleep. Twelve years later, Hans Berger invented electroencephalography (EEG), which was presented in 1929 [13]. The EEG was invented to measure cerebral physiology, especially focusing on mental diseases [157]. Figure 2.3 shows examples of different types of brain waves and patterns that can be found in EEGs.

Polysomnograms, which derives from Greek and Latin (*poly*, many; *somnus*, sleep; *gramma*, drawing or diagram), are used to monitor several functions of the human body during sleep [70]. Among others, in a polysomnogram we can find the following elements:

- **Electroencephalography** measures brain activity.
- **Electrooculogram** records eye movements.
- **Electromyogram** registers muscle activity.
- **Electrocardiogram** records heart rate.
- **A respiratory monitor** measures the respiratory effort.

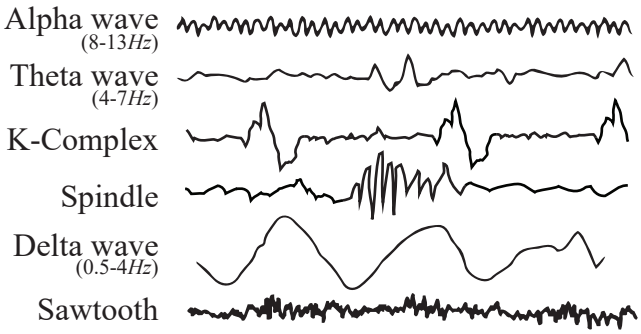
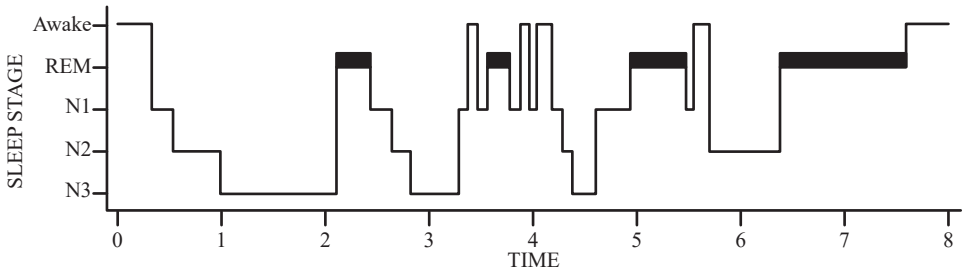


Figure 2.3: Examples of EEG waves. Time and amplitude scales are different in each example.

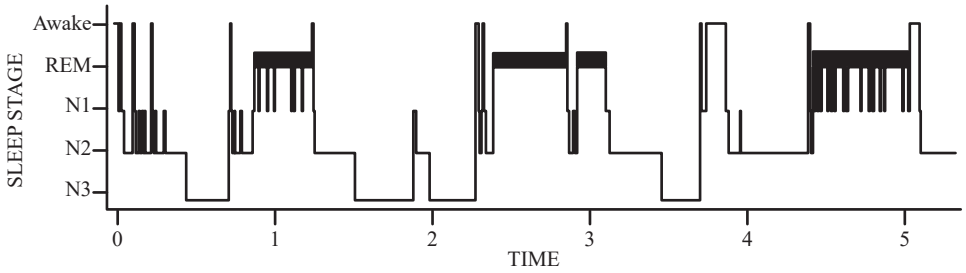
When a patient is believed to be suffering from a sleep disorder, polysomnography (PSG) is often done. PSGs represent an objective method to deliver diagnosis regarding sleep disorders [120]. The PSG is, therefore, usually combined with the clinical history of the patient to make an informed prognosis. The result of a PSG is a polysomnogram. The measurements done in a PSG are used to assess the sleep structure of the patient. Afterward, the PSG is evaluated and annotated on an *epoch-by-epoch* basis. *Epochs* represent a time-fixed period of the night (typically 30 seconds duration), which can then be analyzed by technicians to assign a sleep stage. The process of assigning sleep stages to epochs is called *sleep staging*. The first approach to standardizing the scoring of sleep was proposed by Allan Rechtschaffen and Anthony Kales in 1968 [134], where sleep was scored epoch-by-epoch as one of seven discrete categories (awake, stage 1, stage 2, stage 3, stage 4, stage REM, and movement time). This standard proved to be useful in many cases. However, it was criticized for being too subjective, resulting in high variability in the evaluation of sleep stages, and for being designed mainly for healthy adults [113].

In 2007, the American Academy of Sleep Medicine (AASM) introduced a new standard. The AASM manual included more detailed information for the scoring of sleep, leaving less room for interpretation and thus addressing the main drawback of the previous standard. The AASM manual defined five disjoint sleep stages [119]: *Wakefulness*, *N1*, *N2*, *N3* and *REM*. These sleep stages are characterized by specific physiological properties, which are based on consensus criteria. Figure 2.3 depicts some characteristics that can be observed in the EEG signals of a PSG. *Wakefulness* with the eyes closed is usually characterized by *alpha* waves (8-13*Hz*) in the EEG produced by the occipital lobe of the brain, while sleep stage *N1* often presents *theta* waves (4-7*Hz*). Other stages are characterized by interactions of multiple physiological stimuli that result in the presence of EEG phenomena like k-complexes, spindles, or sawtooth-like waves.

The sequence of annotated sleep stages during sleep is visually represented by a *hypnogram*. Hypnograms are analyzed by somnologists to understand the sleep pattern of a patient. Therefore, they are considered the *de facto* visual representation of sleep stages. An example of a normal hypnogram (i.e., common behavior among healthy subjects) is shown in Figure 2.4a. As we can see, the sleep progresses over the night transitioning between different



(a) Hypnogram of a normal sleep where non-REM stages are more frequent in the first half of the night, whereas in the second half of the night REM sleep is more frequent [62].



(b) Abnormal hypnogram. We can see the sleep fragmentation in the transitions happening between REM and N1.

Figure 2.4: Two examples of hypnograms, where (a) represents a normal sleep behavior and (b) depicts abnormal behavior.

stages. From this visual depiction, new insights can be extracted. For instance, we can observe that the subject follows a normal sleeping cycle (Awake, non-REM, REM). We can also observe some patterns interrupting this cycle. Three hours after the beginning of the sleep, the subject woke up a few times. Figure 2.4b depicts an abnormal hypnogram where the presented sleep pattern is fragmented (i.e., many transitions in a short span).

2.3.4 **Machine Learning in Sleep Staging**

The study of sleep is an important area in medical research. It can reveal disorders, such as apnea, narcolepsy, parasomnia or hypersomnia, which can also relate to other types of medical conditions, such as psychiatric disorders, neurodegenerative diseases [193] or cardiovascular disorders [153]. Therefore, having a good understanding of sleep is crucial to provide better diagnoses of some diseases.

The current procedure to study sleep patterns in clinical settings consists of several steps. First, a PSG is performed to record brain signals, eye, chin and leg movements, blood oxygen level, heart rate and breathing of the patient. After the recordings have been obtained, a PSG technologist determines sleep stages.

Sleep staging is still a time-consuming and subjective process [91]. Scoring an overnight PSG can take from 2 to 4 hours with a scoring agreement of 82% on average between experts

[72]. In the late '60s, some automated techniques were proposed to score the sleep of humans from EEG signals [151, 186]. Since then, many other techniques have been proposed [88, 127, 158, 167, 184]. Other approaches tried to address sleep staging from a different perspective. Instead of providing a fully automated technique, they involve user interaction [4, 28]. Despite good predictive accuracy from these models (e.g., 99% [26]), their usage in medical domains is not yet adopted [27]. This may be due to the lack of trust in models and their somewhat limited ability to generalize to other populations of patients. The latter may be overcome with more heterogeneous datasets. The former, however, requires more sophisticated solutions.

2.4 **Visualization**

Analyzing raw data is an arduous task on its own due to the lack of explicit relationships within the data. Increasing the amount of data, therefore, makes such analysis even harder. Often, in these contexts, statistics are used to gain insights into the raw data. However, statistics do not tell the whole picture and often lack context.

Visualization conveys insights through visual representations that are used to support specific tasks for specific users. Sketches, pictures or graphics can be considered forms of visual representations. In this sense, computers can be used to automatically generate those representations from data to gain insights. Tamara Munzner [114] defines this as "*Computer-based visualization systems provide visual representations of datasets designed to help people carry out tasks more effectively*". A great example of the strength of visualization was provided by Frank Anscombe [9]. In his work, four simple examples were provided showing that statistically equivalent datasets can still have completely different properties. In Figure 2.5 we can see Anscombe's example. Once they are visualized, it becomes immediately clear that these datasets have different characteristics, and shows that combining statistical analysis with visualization can provide valuable insights into the data.

Visualization exploits the ability of the human brain to quickly recognize objects and patterns in visual inputs. The process of visually recognizing objects or patterns is done mainly in two phases. In the first phase, *pre-attentive processing* [165] is done where objects are perceived in parallel. This processing does not require much effort and the whole image can be observed in one go. In contrast, the second phase requires more focus since objects detected in the first phase have to be viewed sequentially.

Data visualization can be used to interpret data. In this sense, two goals can be defined: *exploration* and *presentation*. In the former, the user of the visualization does not know where to look for answers where the latter aims to communicate the answers. Therefore, data presentation is used to help people to communicate results. In this dissertation we focus on data exploration.

Iliinsky et al. [71] indicate data exploration is suitable "*when you have a whole bunch of data and you're not sure what's in it*". Therefore, *exploratory* analysis is needed where answers to some questions may trigger new questions, resulting in discovering nontrivial insights [78]. This type of analysis usually involves *interaction* as defined in the model of Card et al. [21].

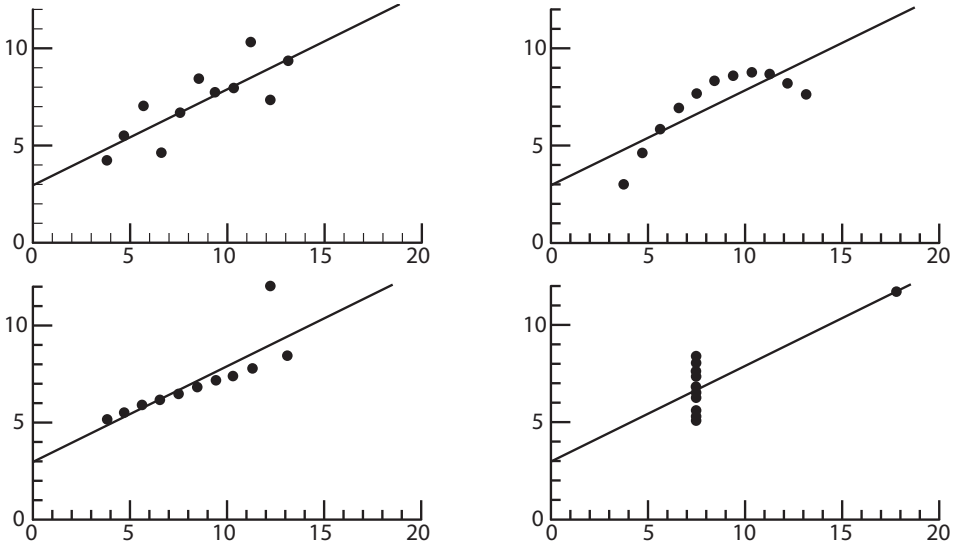


Figure 2.5: Anscombe's illustration [9] of four different datasets which share the same statistical characteristics. In contrast, their visualizations are different, revealing insights that would remain hidden with plain statistics.

Figure 2.6 shows a depiction of their model. As can be seen, interaction can be applied at different stages of the model modifying the transformations or mappings that are involved. The model begins with a data transformation step that filters and aggregates the raw data such that only the information needed for the analysis is kept. After this transformation, visual encodings to visually represent the data are determined. Lastly, the visual representations are presented in the display by applying certain transformations like zoom, distortions, etc.

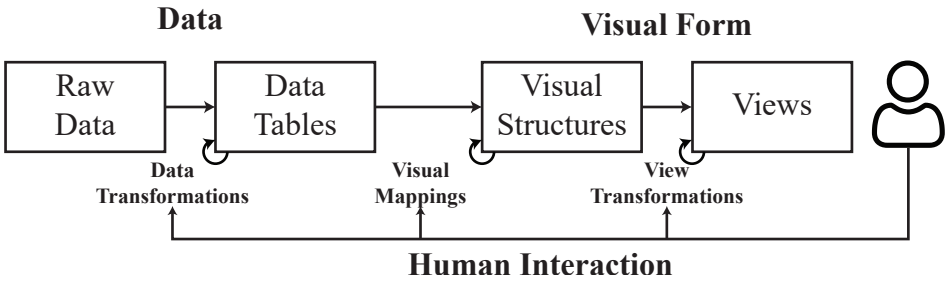


Figure 2.6: Visualization model proposed by Card et al [21]. The process involves two high-level steps: data transformation and visual transformations, which can be altered by human interaction.

Interaction can be divided into subcategories in a data exploration process. According to Yi et al. [195], interaction can be classified based on the intent of the user as:

- *Select*: mark something as interesting. This interaction is usually coupled with other interaction techniques and it seems to work as a preceding action to subsequent ac-

tions.

- *Explore*: show me something else (e.g., panning). Explore interaction is important due to the limitations of available space in the display or perceptual and cognitive capacities of the human brain to process information. Through explore interaction, new data items can be shown.
- *Reconfigure*: show me a different arrangement (e.g., jitter). Revealing hidden characteristics of the data is crucial in data exploration. This can be achieved by changing the spatial arrangement of the items visualized. Moreover, this interaction can also be used to reduce occlusion.
- *Encode*: show me a different representation. Changing the mapping of data to visual appearance, using features such as color, size, etc. serves an important role as it can affect the pre-attentive perception. Interactivity is key to find a proper visual encoding for the data.
- *Abstract/Elaborate*: show me more or less detail (e.g., zooming). Adjusting the level of abstraction plays an important role in how the data is perceived. This interaction aligns with the visualization mantra of Shneiderman [146]: overview first, zoom and filter, then details-on-demand.
- *Filter*: show me something conditionally (e.g., dynamic query). Filter interaction is used for the user to specify certain criteria that changes the set of data items being presented. Usually, the actual data remains unchanged and can be recovered when the filter is reset.
- *Connect*: show me related items (e.g., brushing). This interaction concerns highlighting relationships and showing hidden data items that are connected to certain data of interest.

2.5 **Visual Analytics**

The previous section introduced data visualization and its main characteristics. Data visualization can be subdivided into the following categories [188]: information visualization [155] and scientific visualization [63]. Information visualization is used to analyze abstract data with no direct geometric meaning, scientific visualization focuses on visualization techniques for data that describe physical phenomena.

Both types of data visualization work with an amount of data that is manageable by traditional visual and interaction techniques. Sometimes, these techniques are not enough due to the large size and complexity of datasets. *Visual analytics* combines analysis techniques such as statistical models, machine learning, etc. with interactive visualization to assist analysts in understanding large and complex datasets [79].

Visual analytics can be defined as the science of analytical reasoning facilitated by interactive visual interfaces [29]. Keim et al. [76] defined the goal of visual analytics as to "*gain insight in the problem at hand which is described by vast amounts of scientific, forensic or business*

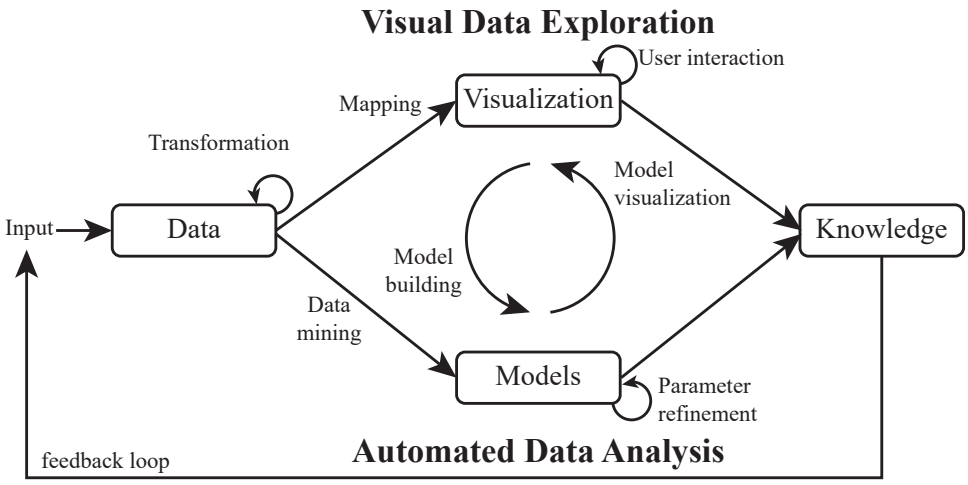


Figure 2.7: Visual Analytics approach depiction by Keim et al. [76, 77]. Data flows into both ML model and visualization system. Interactions occur between both, as well as themselves individually, to provide knowledge that can ultimately give feedback to the initial process.

data from heterogeneous sources. To reach this goal, visual analytics combines the strengths of machines with those of humans". Figure 2.7 shows the data flow of a typical visual analytics system. Data flows into both the model and the visualization system. Synergies can occur triggered by human interaction, which results in knowledge that is used to feedback the process, iterating while necessary.

2.6 **Visualization in Process Mining**

Most work done in process mining does not involve visualization at all. However, in certain situations visualization can play an important role to communicate results. ProM [177] is one of the best-known frameworks in process mining. It includes a wide variety of techniques to mine and analyze process models. A suitable example of how visualization is utilized in ProM to analyze events is the Dotted Chart [154]. Figure 2.8 shows a screenshot of the Dotted Chart applied to an event log for a road traffic fine management process [101]. We can notice, for example, the green vertical stripes, which happen in batches. These events correspond with credit collection, which usually is done in batches. As can be seen, this visual representation can be used to explore the event sequences and gain insights.

A common characteristic among process mining solutions is the ability to visualize node-link graphs. Generally, these graphs depict the process model being analyzed. For instance, Disco [59] presents the process model as a set of nodes and paths connecting these nodes. The model is mined using a fuzzy miner [60], which is meant to avoid *spaghetti-like* models. Furthermore, Disco provides more visual interactions to help the user make sense of the data contained in an event log. Figure 2.9 shows the statistics view where information about events is presented in different views.

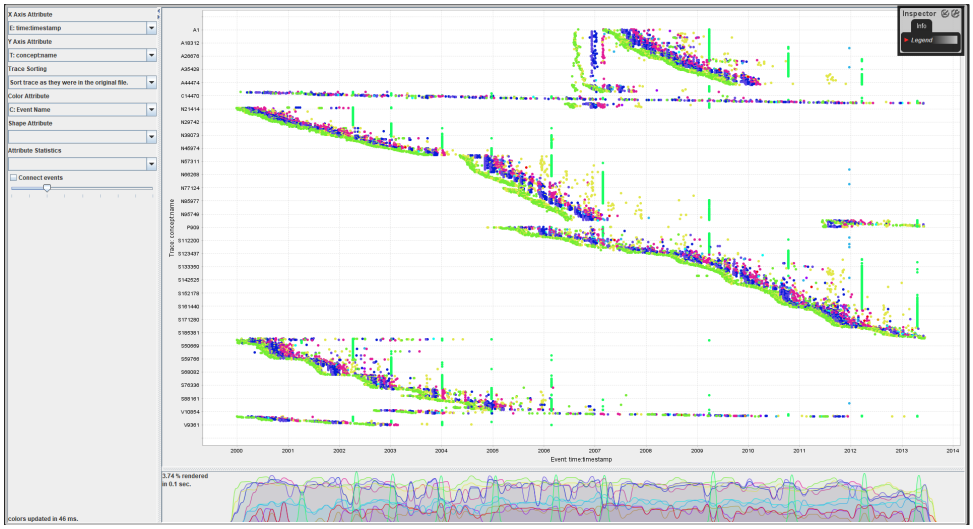


Figure 2.8: Example of the dotted chart [154] generated by the author using ProM [177]. The horizontal axis depicts time, the vertical axis indicates cases and each dot represents an event. Dots are colored accordingly to depict event names. A few options are provided on the left panel to modify the visual encodings of the chart.

Finally, other business software uses more advanced concepts of data visualization where multiple dashboards are presented to the user to better understand their processes. UiPath Process Mining [168] combines multiple views to gain insights into the events contained in an event log. It also enables the user to make changes directly on the process model, providing real-time statistics on performance gain. Figure 2.10 shows a screenshot of their system. We can see that the model depicts the main view. This view can be manipulated to show or hide detailed information. Colors are used to highlight the most often occurring events, while spatial location is used to depict the most common path in the process model.

2.7 **Visualization in Sleep Staging**

Sleep staging is a task in which technicians score the sleep of a patient by visually inspecting different EEG signals. The vast majority of systems present the EEG signals as line charts. Alternatively, other systems also include video recordings of sleep. Both EEG and video are visualized together. This can help the technician to detect certain situations like movements, snores, etc.

Recently, more work has been done in semi-automating sleep scoring. Semi-automated methods complement traditional approaches by detecting certain features in the input data and showing them to the technician. For instance, the work of Combrisson et al. [28] provides a visual system where EEG features (see Figure 2.3) are automatically detected. Such features are color-coded within the line charts. Figure 2.11 shows a screenshot showing some detected features.

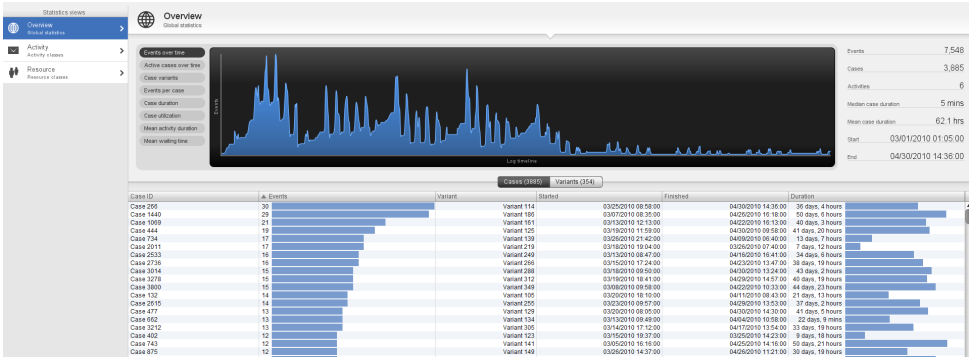


Figure 2.9: Screenshot of the statistics view of Disco [59] generated by the author. The top view features an area chart where the number of events is shown over time. The bottom view shows a table where information per case is displayed. The table uses vertical bar charts to ease comparisons between cases for certain attributes like number of events and duration.

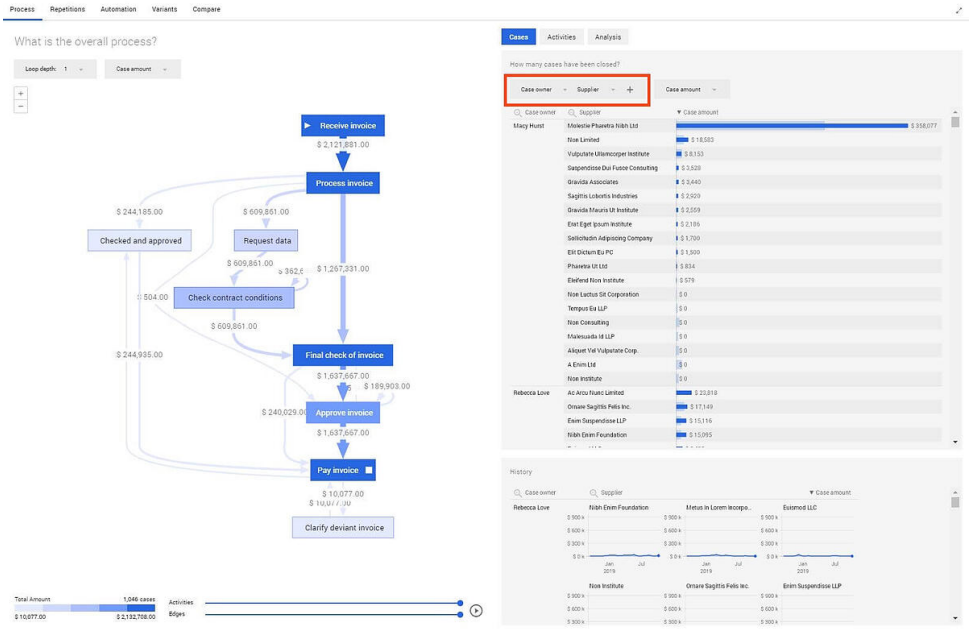


Figure 2.10: Screenshot of UiPath Process Mining [168] taken from [168]. The process model is displayed front and center. Below the model, some controls are provided to alter the detail of the displayed model. On the right several views feature bar charts and line charts to display information regarding cases, activities, etc.

2.8 **Conclusions**

Workflows can be very complex and their tasks can be time-consuming. Analyzing and understanding medical workflows can benefit both patients and practitioners in improv-

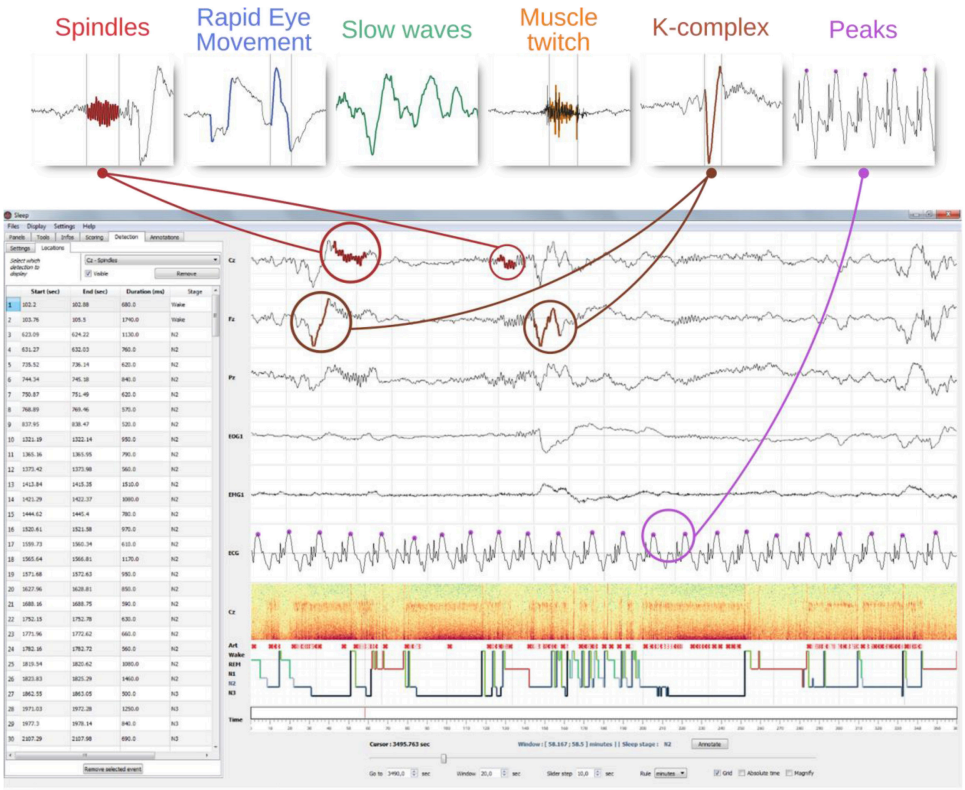


Figure 2.11: Example of semi-automated sleep staging (taken from [28]). EEG features such as spindles, k-complexes, etc. are highlighted in the EGG being visualized to support the user.

ing medical guidelines or reducing the time that certain tasks require. Regarding workflow analysis, current techniques do not enable users to explore the dynamics of models (e.g., soundness property), or combine process-centric approaches with clinical data. This limits the analysis that can be performed. Within a workflow, certain tasks can be optimized by the usage of machine learning. However, this poses new challenges, such as validation of the output data or the performance of the models employed. Visualization can address these problems providing users with new ways to analyze and cope with their workflows and machine learning models. How to effectively apply visualization in these domains remains an open question that will be addressed in the following chapters of this dissertation.

3

Soundness Analysis in Petri nets

Soundness validation of process models is a complex task for process modelers due to all the factors that must be taken into account. Although there are tools to verify this property, they do not provide users with easy information on where soundness starts breaking and under which conditions. Providing insights such as states in which problems occur, involved activities, or paths leading to those states, is crucial for process modelers to better understand why the model is not sound. In this chapter, we address the problem of validating the soundness property of a process model by using a novel visual approach and a new tool called PSVis *(Petri net Soundness Visualization) supporting this approach. The* PSVis *tool aims to guide expert users through the process models in order to get insights into the problems that cause the process to be unsound.*

The contents of this chapter have previously appeared in **Garcia Caballero, H. S., Westenberg, M. A., Verbeek, H. M. W., and van der Aalst, W. M. P.** Visual analytics for soundness verification of process models. In *Business Process Management Workshops* (Cham, 2018), E. Teniente and M. Weidlich, Eds., Springer International Publishing, pp. 744–756 [53].

3.1 Introduction

The use of process models for workflows has been studied for some decades now. It started back in 1979 with the work of Skip Ellis on office automation [41]. Still, it took two decades until notions such as workflow nets and soundness were defined [1] thus linking workflows to Petri nets. As a result of this link, a lot of existing Petri net theory (like [37, 115, 124, 135]) became instantly applicable to the workflow process model domain. Nevertheless, some other approaches to the verification of these models also still emerged, in which was used [139] another graph-like notation for processes in combination with dedicated graph reduction rules.

The application of Petri nets to the workflow domain [2, 3, 106] triggered a new line of research focussing on the soundness verification tool *Woflan* [182], different variations on soundness [34, 180], and extensions to the Petri net formalism (like EPCs [105, 178, 179], BPEL [121], and YAWL [183, 194]). Verbeek et al. [182] also introduced the concepts of the problematic runs (called *sequences* in that paper), which are used in this chapter.

None of these approaches offered a comprehensive visualization of the problems using the process model of choice (YAWL, EPC, Petri net, etc.). For instance, the Woflan tool did not visualize the Petri net it was checking soundness on, but just showed a series of messages that included the labels of the nodes in the net. Nevertheless, the Woflan tool was later included in the process mining framework ProM [175, 179], which did allow this net to be visualized. As a result of this inclusion, selected markings (like deadlock markings) could be visualized by projecting them onto the net, and other Petri-net-related properties (like invariants) could also be visualized. However, such visualization means are still limited and users struggle to diagnose real problems.

This Chapter takes this initial and rudimentary visualization of soundness problems in ProM some steps further by focusing on the visualization. Also, whereas Woflan requires a unique final marking (which should be reached to achieve success in the workflow), the approach in this chapter allows for any collection of such final markings. By visualizing any problem that prevents the workflow from reaching any of these final markings, the user is guided towards correcting the root cause of these problems.

The concept of runs as shown by the visualization is known in the Petri net field and originates from Desel [36]. An example tool that supports these runs is VipTool [12, 38]. VipTool can provide the user with information on whether a given *scenario* (say, a partial trace) fits the Petri net at hand. In this chapter, we assume that such a partial trace fits the net. If not, we can use alignments to find the closest path in the model [172]. This is not supported by the VipTool, but is supported by ProM. We are more interested in visualizing the execution of the fitting partial trace in the net. Apart from this, VipTool can also synthesize a Petri net from a collection of scenarios. This connects VipTool to the field of process mining [175], which is the natural habitat of ProM. However, we do not use that feature of the tool.

PSVis (Petri net Soundness Visualization) is a tool to spot problems in Petri nets through visualization. The tool aims at guiding expert users through the process models to get insight into the problems that cause the process model to be unsound.

3.2 Problem Definition

In this section, we give definitions of the core concepts that our visualization tool needs to handle. We define the problem that we address and present a set of tasks. We designed our visualization tool accordingly.

3.2.1 Problem Analysis

Process modelers tend to ensure the soundness property of the process models. However, it is fairly easy to break this property with only a small number of changes on the model. In addition, models which are derived by discovery algorithms do not always ensure this property. Well-known miners like the Alpha-miner and the Heuristic-miner often produce models that are not sound (e.g., have deadlocks).

Because of the dynamic nature of the behavior of the Petri nets, understanding where the problems occur and the context in which they happen plays a key role for process modelers.

In order to address this problem, we define the following tasks, which form the design basis of our tool:

T1 Obtain an overview of all or a subset of final states of a Petri net.

T2 Compare disjoint sets of transitions and/or places belonging to a specific area.

T3 Find problematic states.

T4 Explore paths that lead to a problematic state.

T5 Determine when the problem occurs for a specific problematic state.

T6 Analyze the runs for a selection of states.

T7 Examine concurrency, loops and causal order in runs.

This set of tasks has been composed in collaboration with experts in the area of process mining and Petri nets to address the problem of soundness validation. To support these tasks, we chose appropriate visual encodings and made an interaction design.

3.2.2 Soundness Validation

We use an algorithm originally proposed by Verbeek [181] to compute the problematic states of a given Petri net. The output of the algorithm is used as input in our tool. The algorithm computes three sets of states, referred to as *Orange*, *Green* and *Red* areas. An abstraction of the resulting output of this algorithm is shown in Figure 3.1.

In our tool, we focus on the border states, that is, the states which depict a transition from the *Orange* area to the either *Green* or *Red* area. In Figure 3.1, the border states are linked by black-dotted arrows. Notice that there may be cases in which the same transition leads to different areas depending on the source state (e.g., *t1* leads to *Green* and *Red* areas). These

states correspond to parts of the Petri net in which everything becomes right (henceforth, all reachable states are *Green*) or everything becomes wrong (henceforth, all reachable states are *Red*).

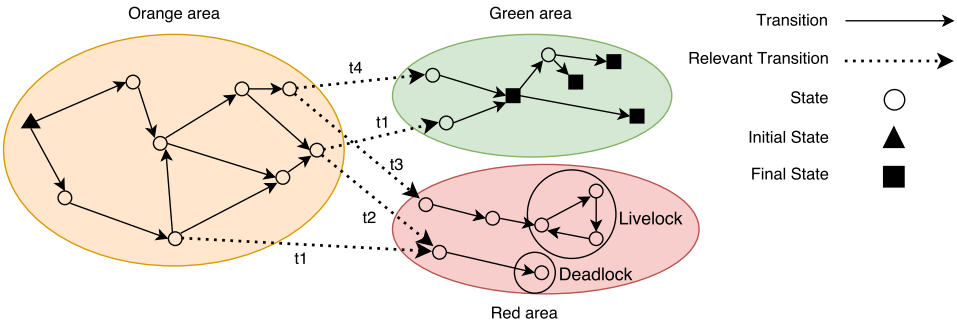


Figure 3.1: Abstract representation of the concept of areas outlined by Verbeek et al. [181]. The state space is divided into three areas: *Orange*, *Green* and *Red*. Circles and arrows depict states of the state space and transitions, respectively. Two possible situations can occur in the *Red* area: deadlocks and livelocks, which are represented by circles without outgoing arrows and cycles.

The *Red* area contains all those states from which is not possible to reach any final state. All reachable states from a *Red* state are *Red* states too. Thus, states within this area are considered as wrong states. On the other hand, the *Green* area represents those states from which a final state can always be reached. Similar to the *Red* states, from a *Green* state only *Green* states can be reached. Lastly, the *Orange* area comprises states from which some *Red*, *Green* and *Orange* states can be reached.

Within the *Red* area, two major problems can be present: deadlocks and livelocks. In Figure 3.1, two problems are indicated with circles. Deadlocks occur when a state from which no transition can be fired is reached, and it is not a final state. This can be seen in Figure 3.1 as the state has no outgoing edges, that is, no transition can be fired at that state. Livelocks happen when a cycle is found, which means that it is possible to iterate between a subset of states forever.

3.3 PSVis

In this section, we introduce *PSVis*, its components, and how they interact to execute the tasks depicted in Section 3.2.1. An overview of all components is given in Figure 3.3. Every component enables to perform a specific task or a set of tasks.

Our tool assumes that the state space is computable in a reasonable amount of time. If the state space contains unbounded places, it is infinite in size and cannot be computed. But even if all places are bounded, the state space may still be too big to be constructed within a reasonable time [182]. To avoid having to spend unreasonable time in constructing the state space, our tool uses a threshold that operates on the number of tokens in a state. If the threshold is set to b, then only states where every place contains less than b tokens are

expanded. This is related to the notion of b-boundedness introduced earlier. Thus, if all places are b-bounded in a net, then setting the threshold to b or higher does not change the state space. If the threshold is reached at some state, we assume that state is a problematic one.

3.3.1 **Glyphs on the Petri net**

In order to support T1 (Obtain an overview of all or a subset of final states of a Petri net), process modelers need to visualize the number of tokens of a specific place, and the states that place belongs to. As a result, we introduce *glyphs*, which decorate places in a Petri net. Glyphs are visual representations of a piece of the data where visual attributes are dictated by data attributes [187].

The number of tokens in a place is represented as dot shapes contained in the place, numbers, or a combination of both. The main problem of this representation arises when we want to visualize more than one final state at the same time. A final state according to the definition is a multiset of places. This implies that a specific place can belong to more than one final state. With the current way of presenting this information, expert users are not able to know the number of tokens contained inside each place for each state and whether a place belongs to more than one state. Therefore, we propose a new way to visually encode this information. This new encoding is shown in Figure 3.4(b).

In our approach, the state with the highest number of unique places determines the color of the places. Glyphs are then colored with the remaining states. The initial state constitutes an exception and the places that belong to it are always colored accordingly. Figure 3.2 shows an example of how our approach works. When the user selects *State 1*, the two places that belong to it are colored red. Next, the user selects *State 2* which has three unique places. Therefore, all the places are colored blue and two glyphs are created to present *State 1*. Lastly, when the user selects the initial state, two places are colored green and glyphs are created to show the remaining states.

If there is more than one token in a place, a label is attached to the glyph (or to the place) indicating the number of tokens. This label is colored dynamically depending on the brightness of the background color. Thus, labels can be colored black or white to make them readable.

3.3.2 **Petri Net View**

This is the main view of our tool. It presents the Petri net (see Figure 3.3(1)) where circles and rectangles depict places and transitions. We implement a version of Sugiyama's approach [162] to layout the Petri net since it gives a good understanding of the flow of the process. Some parameters of the layout algorithm can be modified through the toolbar at the top of the view. In addition, users can perform zooming, panning and dragging of elements directly on the view. Last but not least, nodes can be hovered to show the label of such elements by using a tooltip.

To the right of the Petri net view, there is a panel that shows the initial and final states of the model. Each state is presented in the tool by two components: a button and a colored rectangle, which are interactive. This component can be seen in Figure 3.4(a). The example shows one single initial and five final states. Users can (de)select states with this component and change the assigned colors. When a state is selected, the Petri net view reacts by showing the current selection of states. Given the fact that a single place can be present in multiple states, we use a new approach to represent that a place belongs to several states (see Figure 3.2). This feature directly relates to task T1 (Obtain an overview of all or a subset of final states of a Petri net).

The top part of this view (Figure 3.3(1.a)) shows two sets of buttons that are dedicated to performing a quick exploration of the problems that have been detected. These buttons enable users to explore places or transitions that belong to just one area. This is useful because it gives a quick overview of the parts of the Petri net that belong to the *Green*/*Red*/*Orange* area. In order to do this, users can select what they want to visualize (places/transitions) and which area they want to explore (*Green*/*Red*/*Orange*). Once the user selects one of these options, the Petri net view highlights those elements which belong to the selected area. This feature relates to task T2 (Compare disjoint sets of transitions and/or places belonging to a specific area).

3.3.3 **States View**

This component can be seen in more detail in Figure 3.4(c) and it enables the exploration of the most important problematic states within the state space of the Petri net (T3 (Find problematic states)). Those states correspond to scenarios in which the process can lead from an *Orange* state to either a *Green* or *Red* state. This component only shows the border cases, which are derived from the relevant transitions of the state space (see Figure 3.1).

Our approach uses a two-level tree to visualize the different states in which the net experiences a problem. The first indicates the states in which an *Orange-to-Red* or *Orange-to-Green*

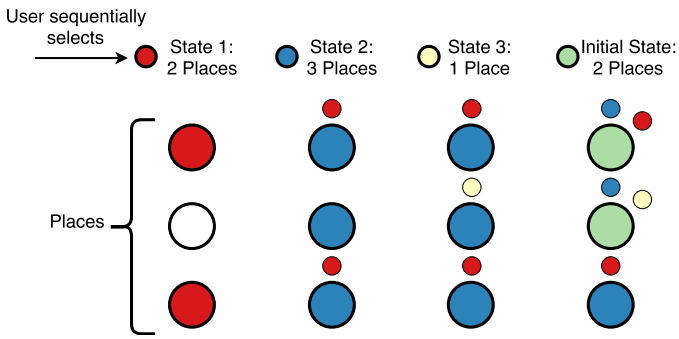


Figure 3.2: Assignment of colors to places when selecting states. From left to right, the evolution of the coloring of the places when a user selects states is shown. Glyphs decorating the places of the Petri net are created on demand.

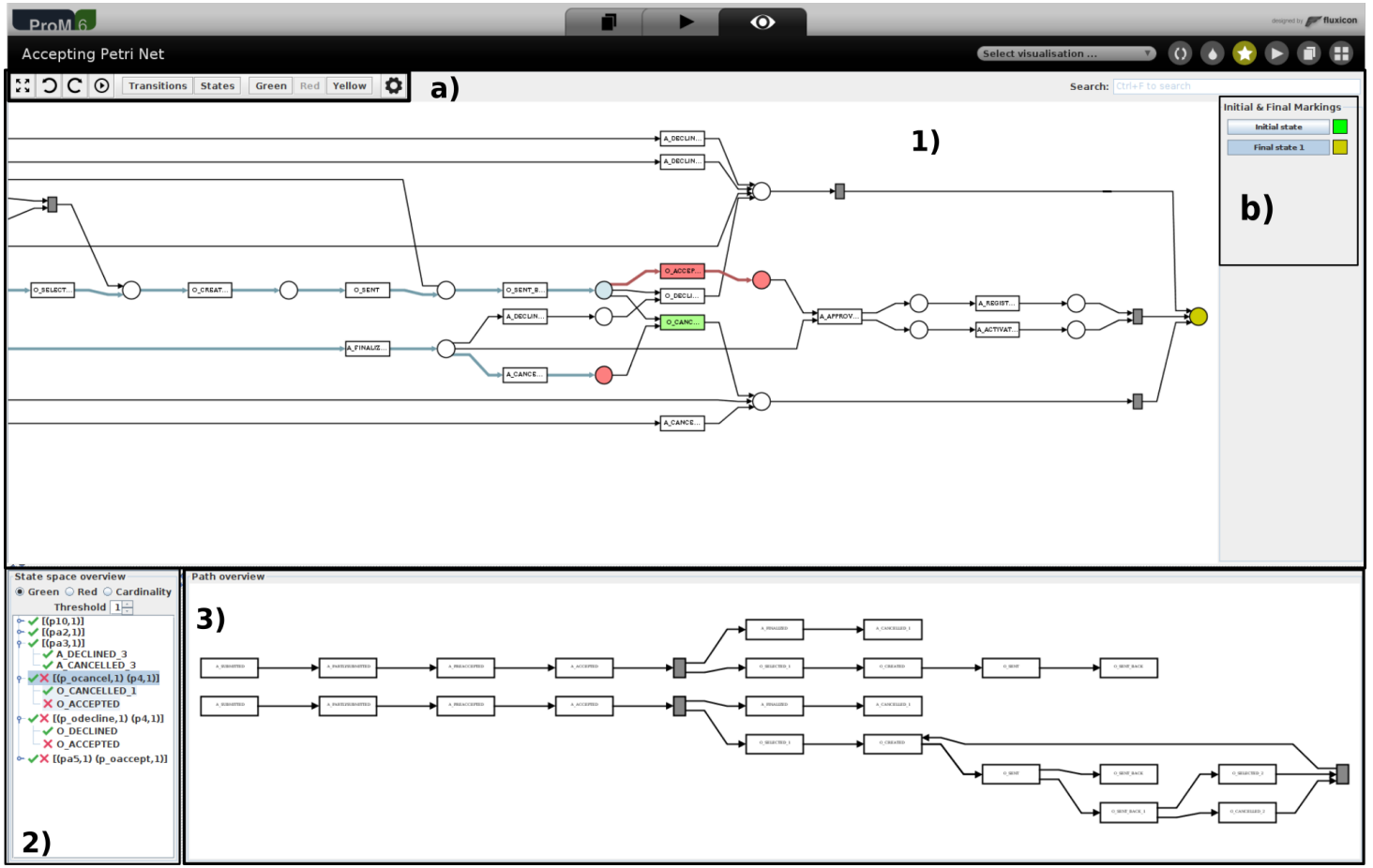


Figure 3.3: Overview of our tool. 1) displays a Petri net in which four places are colored: two in red, indicating the last state reached in the *Red* area, one in blue, indicating that token was consumed to reach the *Red* state, and one in golden, indicating the final state of the Petri net, which was selected in 1.b). 2) shows all the available problematic states, some of them are highlighted indicating that the user selected those. Lastly, 3) shows the corresponding runs.

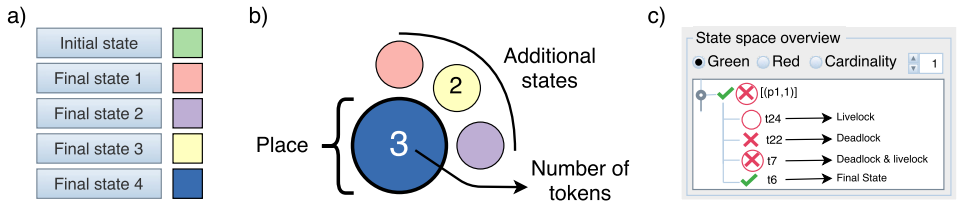


Figure 3.4: Examples of the components of our tool: a) component showing the initial and final states and their colors; b) four different final states that involve the same place; d) component showing a problematic state from which we can reach four different types of problems depending on the transition we fire.

scenario was found, and the second level indicates the transition that is involved in the detected scenarios. For each state, the user can find a variable number of transitions that could trigger a step in the state space leading from an *Orange* state to a *Green* state, or a *Red* state in which two situations can ultimately happen: deadlock or livelock. This part relates to T5 (Determine when the problem occurs for a specific problematic state) since they show where (state and transition) problems occur.

In some cases, the algorithm that our tool uses detects a problem only because of the threshold that it uses. In those cases, this component indicates that by marking the problematic state with an asterisk. In this way, users can easily differentiate those states that are always problematic from those that are problematic because the algorithm did not continue exploring.

We use icons to give a quick overview of the scenarios found by the algorithm. They indicate the scenario for a specific state (aggregated view) and a specific transition (specific view). Therefore, the first level of the tree might display multiple icons indicating all the possible scenarios that can be reached through that state. Seven scenarios are possible: 1) reaching a *Green* state; 2) a *Red* state that eventually leads to a deadlock is reached; 3) a *Red* state that eventually leads to a livelock is reached; 4) a *Red* state from which a deadlock and livelock can be reached; 5) either a *Green* state or a *Red* deadlocking state can be reached; 6) either a *Green* or a *Red* livelocking state can be reached; and 7) either a *Green* or *Red* dead/live-locking state can be reached.

The nodes of the tree can be sorted by three different criteria. By clicking on the corresponding radio buttons, the view sorts the states by the criterion chosen by the user. Thus, users can sort states by the type of scenario that they represent (either *Green* or *Red*) and the cardinality of the states.

Next to the sorting functions, there is a spinner, which is used to set the threshold used by the algorithm that computes the state space. By default, this parameter is set to 1. When the user interacts with the spinner, the tool recomputes the state space, partitions the recomputed state space into *Green*, *Red* and *Orange*, and recomputes all the relevant information related to them, such as runs or disjoint sets of states and transitions.

Users can interact with the nodes of the tree to explore the different scenarios. This way, the nodes can be selected to be displayed in the Petri net view. When a node from the first level

of the tree is selected, the main view shows all the available scenarios for that specific state by coloring the nodes of the Petri net that are involved in that specific state. In Figure 3.3 state *{p_ocancel, p4}* is selected. The places that define the selected state are colored blue, while the transitions that can be triggered leading the process to a *Green* or *Red* state are colored green and red.

Once users select a (set of) state(s), it is possible to interact with the main view to explore the behavior of the net. This is done by enabling users to click on the transitions that have been colored to show the paths that lead to the selected state, and the final marking reached by triggering that transition. This feature connects this view and the runs view, which is described below.

3.3.4 **Runs View**

This view helps users perform tasks T4 (Explore paths that lead to a problematic state) and T6 (Analyze the runs for a selection of states). An example is shown in Figure 3.3(3). Runs are displayed as disconnected graphs, which can be projected as paths in the Petri net view. When users select a state from the *states view*, this component shows the runs that lead to the chosen states. Then, two major interactions are provided: nodes hovering, and path selection. On the one hand, the first interaction aids users in linking nodes of the runs to nodes of the Petri net view. On the other hand, the second interaction assists in visualizing the path that goes from the initial state to the selected state directly on the Petri net.

Through these two interactions, users can detect states that share similar segments of the path, helping users get insights into the problematic scenarios of the model.

There is a third interaction that links directly with the Petri net view. When the user clicks on a transition that is involved in the problematic scenario that the user is exploring, the Petri net view takes the run that leads to the ultimate state reachable from the current state, that is, a deadlock or livelock, and shows the path. Providing the context in which the process ends up in a *Red* state helps the user to understand how the process led to that problem.

3.3.5 **Design Decisions**

The design decisions made in this work support some basic notions on human perception [111]. The usage of glyphs to represent the belonging of a place to different states is a natural way to show that type of information. They are located next to the elements for which they provide information, and they use a basic color code to show the state that they represent. We use this notation since it is known that the color is a cognitively effective visual variable [111]. Also, we use colors to display useful information on top of the Petri net. We consider this approach to be acceptable since the usage of other ones (e.g., shapes) would probably interfere with the Petri net notation itself. Even though color representations are a limitation for our tool, we consider them satisfactory for this work.

3.3.6 **Implementation Details**

Our tool is implemented as a plug-in within the ProM [177] framework using Java v6. Prefuse [64] is used to manage the graph structures and the visual properties of the visualization of the Petri net. The jBPT library [130] for Petri nets is used to compute the runs.

3.4 **Use Cases**

In this section, we demonstrate how our tool can be used to assess Petri nets. We focus on two nets, which were designed by students who participated in the study developed in [182]. These nets include information on initial and final states.

The first case exposes an example in which no final state can be reached. Figure 3.5 summarizes this use case. It shows an overview of the Petri net and an area of interest in more detail. As can be seen, there are no problematic states in the states view. However, we observe that some of the places are colored red, which means that this place can contain tokens but that no final state can be reached from any state in which this place contains tokens, and some are not colored at all. The latter places do not appear in the state space, which means they are not reachable since if they were reachable, they would be either colored or present in the problematic states view. The final state is also not colored, and therefore the process can never reach the final state. Observe that all three input places of the *book_hotel_...* transition can contain tokens, but the single output place cannot. Apparently, not all input places of this transition can contain tokens at the same time. The source of this problem can be found in the second place from the left in the overall net, which corresponds to a three-way choice. If the highlighted path is chosen, then only one of the input places can contain a token. Otherwise, only the other two can contain tokens.

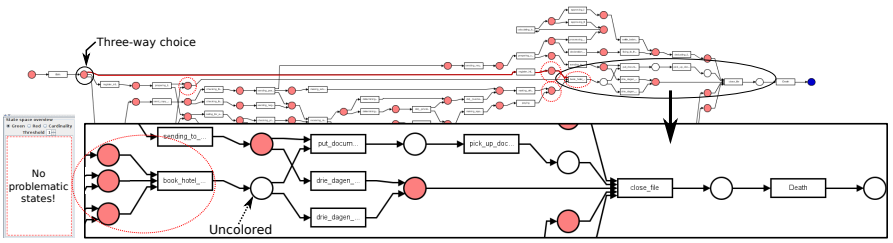


Figure 3.5: Screenshots of the usage of the tool for a dataset.

The second use case (Figure 3.6) depicts a Petri net in which several problematic scenarios have been found. Initially, we proceed by exploring the disjoint sets of places and transitions. We can easily spot some straightforward paths that lead to the final state as well as some paths that eventually finish in the *Red* area (Figure 3.6(a)). Furthermore, we see some places and transitions that have not been colored. We now focus on those elements to explore what occurs there. By hovering on some of the places that have not been colored, we can see their labels. One interesting place is the one labeled as *Status7* since it is present in three of the

problematic states shown in the states view. We pick one of those states to explore the runs that lead to that problematic state. The tool shows two different runs (see Figure 3.6(a)). By looking at these, we can see that they share some of the initial steps in the process, but that they divert at some point. Clicking a run visualizes the paths that those runs represent in the Petri net. If we compare the two runs in this way, we immediately see that they finish in the same problematic state, although they followed different paths. We can also see where the deadlock occurs by clicking on the transition colored red. Figure 3.6(b) shows the two paths (edges colored blue) for the two runs, as well as the deadlock that is reached (place colored red). The place colored blue indicates that the token must be consumed by firing the red transition.

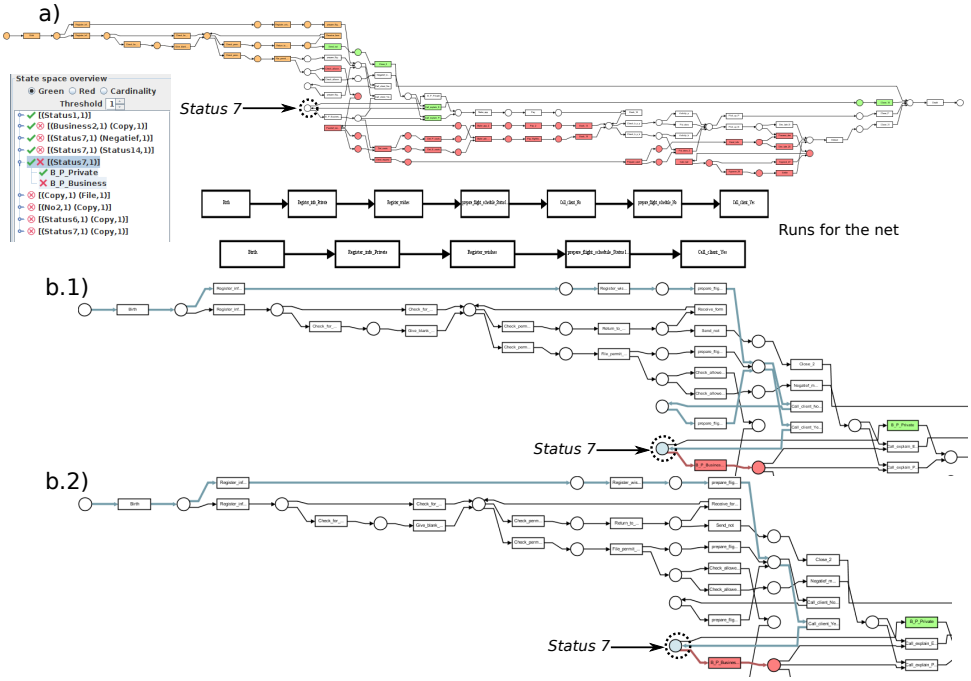


Figure 3.6: Screenshots of the usage of the tool for a dataset.

3.5 **Conclusion**

We have presented *PSVis*, a tool to visually assess the soundness of a Petri net. We have formulated the most important analysis tasks, and have demonstrated the usage of our tool through the exploration of two Petri nets. The first use case showed a simple scenario in which we discovered why the final state was not reachable. The second use case represented a more complex example in which more actions were performed. Through different actions, we observed different aspects of the Petri net such as deadlocks and common paths that lead to them.

One of the main limitations of our tool is that it relies on the state space, which cannot always be computed in a reasonable time. Even though we have some workarounds (setting a threshold to limit the computation of branches), it may take a considerable amount of time to finish. We plan to study alternatives to compute the parts of the state space that are used in the analysis. One option might be to explore the state space incrementally by computing just portions of it.

4

Performance and Conformance Checking for Process Models

Digital platforms in healthcare institutions enable tracking and recording of patient care pathways. Besides the Electronic Health Records (EHRs), the event logs from Hospital Information Systems (HIS) are a very efficient source of information, from a both operational and clinical point of view. Process mining allows comparison of a patient care pathway with the event log(s) from HIS, to understand how well the reality as depicted in the event log fits the expectation as modeled using a care pathway. In this chapter, we present SepVis, a visual analytics tool that aims to fill the gap in current process-centric applications by looking at patients' pathways from a clinical point of view. We demonstrate the utility of SepVis in selected use cases derived by the guidelines in the management of sepsis patients.

The contents of this chapter have previously appeared in **Garcia Caballero, H. S., Corvò, A., Dixit, P. M., and Westenberg, M. A.** Visual analytics for evaluating clinical pathways. In *2017 IEEE Workshop on Visual Analytics in Healthcare (VAHC)* (2017), pp. 39–46 [50].

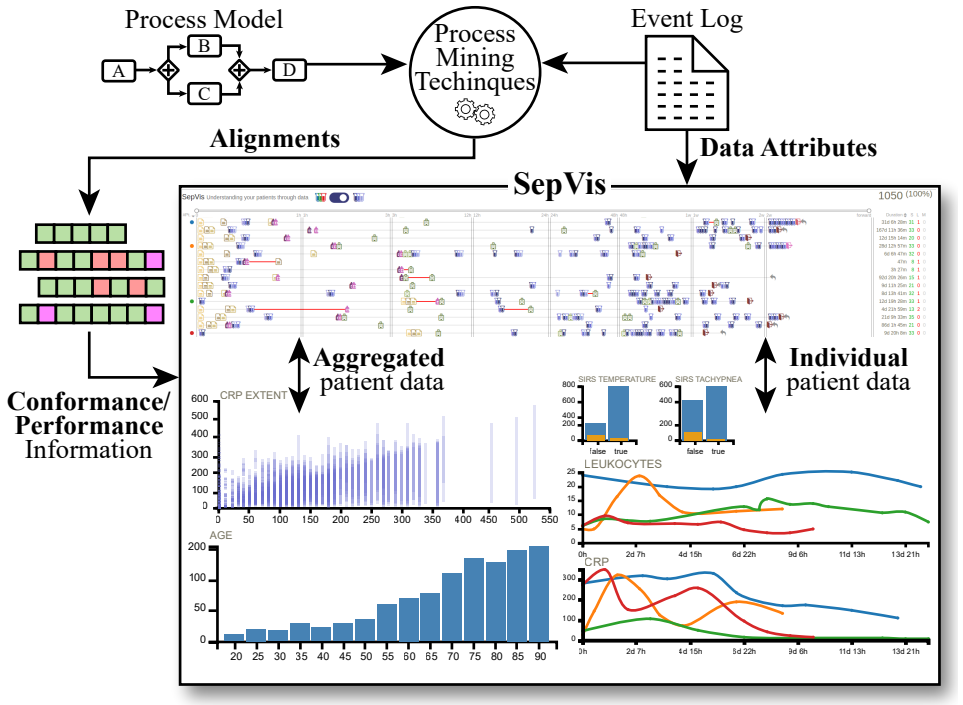


Figure 4.1: Overview of SepVis. A visual analytics tool that takes performance and conformance information from alignments. It provides different views to explore clinical pathways as well as patient-related data. The data attributes are extracted from an event log. A process model describing a guideline in a hospital setting and the event log are used to compute the alignments by an algorithm.

4.1 Introduction

A visit of a patient to a hospital results in a certain sequence of activities that are performed by specific resources. Some activities, such as administrative tasks, could be valid for most of the incoming patients. However, some other activities are specific to a cohort of patients, depending on factors such as the type of disease, diagnosis treatment plan, etc. Generally, a hospital has a well-defined care pathway in place depending on the type of incoming patient. These patients are the primary actors (cases) of the care pathways, which inherently consider the essential protocols and/or guidelines.

With the advent of digitization, most of the steps performed, such as registration of ward activities by clinical personnel, treatment plans and results, logistics and medical decisions, are well documented in the Hospital Information Systems (HIS) [164]. For a limited number of patients in a hospital, the data from the information systems can be retrospectively explored to analyze clinical decisions, deviations from the care pathways, etc. However, the task of analyzing the data from the HIS in a care pathway setting can be extremely challenging if the number of patients is large. This is especially the case for a single type of ward

and/or disease because patients may have different profiles and undergo different *versions* of the care pathway. This adds new dimensions, and hence complexity, to the data and makes the orchestration and integration of data challenging. However, this additional information also enriches the value of tracked activities in the patients' flows, by providing additional attribute information.

One key enabler to the analysis of huge, care pathway-specific data is Process Mining [171]. Traditionally process mining is divided into process discovery and conformance analysis. Process discovery, as the name suggests, discovers a process (care pathway) model from any event log, which is typically derived from HIS or EHRs in a healthcare setting. Conformance analysis uses a process model and projects an event log on the process model [172]. Conformance analysis provides insights into which part of the process fits the data, and which part does not, thereby giving insights into deviations that occur compared to the expected behavior. Furthermore, process mining output can also shed light on performance-related information, such as throughput time of a patient in a care pathway setting, bottlenecks in the process, slow paths in the process, etc. This information could be used to assess the performances in hospital [145] settings and/or to cut down costs and delays. Root cause analysis could be performed on the computed process information to investigate the reasons for deviations and reasons behind different decisions.

In the analysis of event logs from HIS, most of the current approaches do not consider the *care pathway* dimension. Instead, the focus is usually on aligning sequences of variants guided by sequence mining algorithms. However, such techniques often bypass the importance of care processes and the clinical protocols and guidelines that are modeled in such pathways. On the other hand, process mining centered approaches take into account these models, but the visualization focus is entirely addressed from a *process* perspective. That is, the outcome is consolidated and projected on a process model. This makes it difficult to understand and visualize patterns or cohorts of patients that may exist. By focusing entirely on the process dimension, other relevant patient attributes that may be important are also not visualized.

In this chapter, we present our work in progress on SepVis, a visualization tool that aims to extend the standard visual analytics tools in the healthcare domain with the support of process mining techniques. Instead of using traditional sequence alignment techniques, which do not consider the important *care pathway* dimension, we use the results from process mining techniques to layout and visualize the activities of patient flows in accordance with the *clinical model*. Furthermore, instead of using a process model, by visualizing the activities directly, a transparent and direct *view on data* is provided for the user to interactively explore and gain valuable insights into the data.

4.2 **Related work**

In recent years, many applications have been introduced within the visual analytics community for the exploration of temporal event sequences and healthcare data. We distinguish two categories in this domain: sequence alignment-based techniques and process analytic

based techniques. In this section, we highlight the relevant contributions from the literature belonging to each category.

4.2.1 **Sequence alignment-based techniques**

Many techniques in the literature focus on visualizing the alignment of sequences to represent temporal event data. The initial work in this category provided tools showcasing a representative description of a single patient record [32, 128]. However, the increasing size of HIS records has boosted the development of tools with a focus on exploring, querying and visualizing large population data. Recent advances in this category are applications such as LifeFlow[191] and OutFlow [190], dedicated to the exploration of event sequences in EHRs. EventFlow [110] primarily focuses on medical data, but it has been applied across many domains, e.g. log-analysis, cyber-security, sports analytics, learning analytics and incident management [109]. In recent work, other visualization approaches for representing pathways within a hospital have been investigated. For instance, a tracking graph construction [189] has been used to describe patient progression over time in an IC unit. Another enabler in finding patterns in longitudinal clinical data is given by interaction which is effectively demonstrated in Borland et al. [16].

However, all of these applications rely entirely on the data from EHRs. These techniques give a powerful view of the reality, used to visualize sequences using algorithms such as sequence clustering. However, contrary to our approach, these techniques do not take process-centric (care-pathway) information into account. Hence, deviations from care-pathway models and clinical protocols cannot be analyzed.

4.2.2 **Process analytic based techniques**

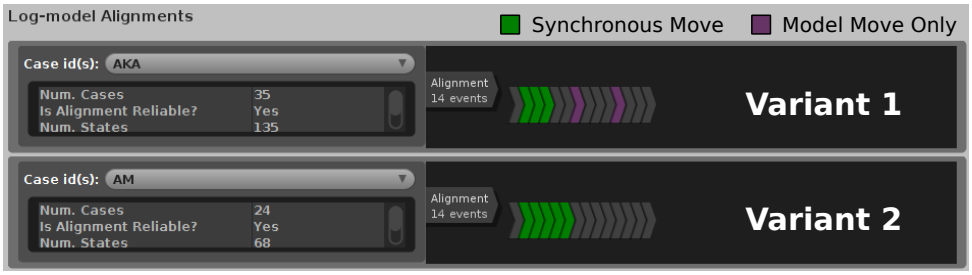


Figure 4.2: Visualization of alignments in ProM. Patients are grouped by variants. The outputs from conformance checking are visible on the variants as colored arrowheads. However, this view does not include information regarding the duration of individual traces, the time between activities and it does not allow the user to explore the process by activities' attributes filtering.

Conformance analysis is a well-researched field in the area of process mining, which extracts valuable information from event logs based on certain process models. Among the commercial tools, Celonis [23] provides visual interaction with process-centric data and enables the

user to define filters according to the available data in the dataset. The process mining framework ProM [177] contains various plug-ins that provide various interesting visualizations of the results from conformance analysis. Figure 4.2 provides a snapshot of one such visualization from ProM. From a visualization perspective, many of the current process analytic techniques project the conformance and performance results directly on a process model. There are a couple of issues in such a visualization practice. First, the modeling notation used for representing the process model could be non-trivial. Hence, for a user, e.g. a doctor, the projected information on the process model might be difficult to comprehend. Second, visual analytics features to investigate and make use of other attributes are often missing. In our previous work [39], which is implemented as a plugin in ProM, we combined conformance and performance analysis with interactive visualization of the process model by explicitly making use of data attributes. However, in SepVis, we aim to make a step forward in the support of visual analytics techniques, by extending the current visualization from ProM for conformance and analysis. Instead of interacting with the process models, which could be difficult to understand to the end-user, the focus in SepVis is on providing a better user experience for the exploration of variants, traces, and deviations using information from the conformance analysis techniques, thereby taking into account protocols and guidelines that form a part of the care pathway. In essence, SepVis combines the visualization of sequence variants with the capabilities of process mining.

4.3 **Dataset and Tasks**

For the purposes of this work, we use a dataset obtained by an Enterprise Resource Planning (ERP) system in the emergency department of a hospital in The Netherlands [100]. Patients suspected of sepsis were tracked from the emergency room, where the doctors diagnose the condition. According to the severity of the disease, indicated by symptoms such as infections, body temperature, respiratory rate, heart rate and diffuse psyco-physic pain, patients are administered antibiotics and directed to normal care (NC) units or intensive care (IC) units [99].

Protocol procedures for sepsis treatment are defined [35], but depending on resources and availability of staff and wards, the care pathways can still vary in terms of scheduling and resources.

The dataset comprises 1050 cases described by 15000 events that were recorded for 16 different activities. In addition, 39 data attributes were included to enrich the pathways of patients. These attributes are divided into two groups: lab tests and information from checklists filled out by nurses. Among the lab tests, we have C-Reactive Protein (CRP), White Blood Cells (Leukocytes) and Lactic Acid tests. These tests are used by doctors to monitor the status of sepsis patients. An interesting fact regarding these tests is that they can occur many times in a single case, and hence our tool has to cope with it. Besides these attributes, we also have attributes indicating the age of the patient, and boolean attributes signaling conditions of the patient. These clinical attributes denote whether the patient matches the Systemic Inflammatory Response Syndrome (SIRS) criteria [10] and organ malfunction, emo-dynamics and inflammatory parameters, which are used in the *sepsis screening triage*. Most of these

attributes were recorded at the emergency room entrance. Next, they are used by doctors to plan the specific care pathway.

The activities contained in the dataset belong to different categories. Some activities are related to blood tests, others concern admission to NC or IC units within the hospital and others relate to administrative tasks etc. Interesting records regarding the administration of *intravenous liquids* and *intravenous antibiotics* are also available.

A set of questions was obtained by interviewing clinicians responsible for the emergency department [100]. These questions arise when the kind of data described above is available on the system. Next, we defined a set of requirements that we used for the implementation and generalization of our tool. Each task is related to specific variables from the process-point of view:

T1 Patients' *activities* are strongly related to patients' clinical conditions *(attributes)*. Users want to be able to see processes from a clinical point of view.

T2 Time between activities hints at the observance of protocols or not. Delays in relation to guidelines indicate the *performance* of the clinical department.

T3 Evidence for changes in care pathways leads to understanding reasons for mistakes, different clinical decisions or inattentive activity recording. *Deviations* need to be visible and clear.

In **T1**, the clinical staff intends to get an overview of recorded cases from the activity perspective. These can be the admissions to a specific ward, a sequence of activities, medications, or laboratory tests. The link between clinical conditions and activities is highly important to understand decisions within the single care-pathway as, for instance, critical patients who are expected to be treated promptly according to clinical protocols.

Task **T2** is performed to look at performances on wards and execution time between activities. In clinical settings, time represents a key aspect and the insight into delays due to unavailability of resources is decisive in management analysis of life-threatening diseases.

Lastly, **T3** is important because clinical procedures in the hospital are susceptible to doctors' decisions. Groups of patients can deviate from the expected behavior because of doctor decisions based on demographic information or clinical conditions (e.g., age, blood tests) that differ from the model but guaranteed success or seemed reasonable at that point in time.

4.4 Design Decisions

Our system SepVis was developed according to the tasks presented in Section 4.3. In this section, we discuss our design decisions.

4.4.1 Icon design

We identified five main categories of activities within our dataset, which we represent by icons. The categories are administration, tests, ward admissions, treatments, and release/return actions. The shapes of the icons match the meaning of these categories. For instance, the icon used to display tests is a test tube, the icon showing treatments is a jar, and so on. This way, users can easily identify groups of activities, and have an idea what is happening to the patient.

Color palette. To assign a color to each activity, we offer two different approaches in our tool. The color palette can be switched easily and the mechanism to do so is explained in Section 4.5.1. We consider that colors can play an important role in two different situations. On the one hand, they can help to identify groups of activities without focusing on the activity itself. On the other, they can be used to spot specific activities more easily. In the first case, the color palette is divided by categories of activities, assigning a color to each category and a specific tone of that color to each activity within the category. In the second case, we use a qualitative scheme of colors. In this way we allow users to quickly find activities of interest since they all have different colors. Both color palettes can be seen in Figure 4.3.

Shifting & opaqueness. Because several events can happen at the same point in time causing overlap of icons, we shift and make icons opaque. Shifting helps to avoid two icons from being placed at the same position. In our tool, we apply a small constant shifting to all icons. In this way, we ensure that no full overlap occurs between two consecutive icons. On the other hand, opaqueness ensures that the icon placed on top of another one is always visible. This helps to disambiguate overlapping icons resulting in a better appreciation of them and their distribution over time.

Border. In some cases, the border of the icons is used to present a special situation such as a model move. Recall that a model move is a situation in which something should have happened in reality, but it did not. This is important since it can reflect a missing step in a process, e.g., the administration of a medicine or the request of a laboratory test. We use this codification since it does not clutter the view excessively and the model moves can still be seen.

Figure 4.3: Icons and color palettes that can be found in SepVis. The first palette is intended to facilitate users to spot individual activities. On the other hand, the second one aims to enable users to identify groups of activities.

4.4.2 **Activity links design**

The activities within a trace are considered to occur sequentially in time. In order to reflect this explicitly, our tool uses links between activities. Besides this, those links can be used to encode some extra information. In our approach, links can represent three different states (see Figure 4.4):

Synchronous moves. This is the normal and most common case. It reflects the link between two activities that happened both in reality and according to the process model. This type of link is presented in our tool by a straight line with a grey color connecting two activities.

Log moves. These moves reflect an abnormal behavior that was recorded in the event log and does not match the process model. We use a red, thicker line to display this type of move. Since the line links two activities, it is colored depending on the type of the latter activity. If that activity is marked as a log move, then the line will be colored as red.

Model moves. The last type of move depicts a case in which something should have happened according to the process model, but it did not. A dotted line is used in this case. The reason behind this choice is that we try to highlight the fact that we do not know if this activity happened in reality and it was not recorded, or that it did not occur.

Figure 4.4: Three types of lines encoding three different types of moves: synchronous moves, log moves and model moves, respectively from top to bottom.

4.5 **SepVis**

In this section, we describe SepVis by explaining its components and how they interact. Our approach consists of clinical pathways exploration, filtering and attribute selection. An overview of our approach can be seen in Figure 4.5. Each view interacts with the others to provide an intuitive perception of patients' pathways to support the tasks **T1**-**T3**.

From the task analysis, we derived the core components of SepVis. An overview of pathways of the patients, as well as the most and the least frequent patterns **(T1)** over time, are given in our main view, the *Pathways view*. For a life-threatening condition such as sepsis, diagnosis time and treatment occurrence play a crucial role in patients' survival. Hence, time and filtering options by activity/pathways duration need to be provided for a proper comprehension of patient management and resource efficiency **(T2)**. Time information is encoded in the *Pathways view* and data can be filtered from the *Performance search*. Clinical data **(T3)**, such as recorded conditions when the patients enter the hospital, the age and the exams they

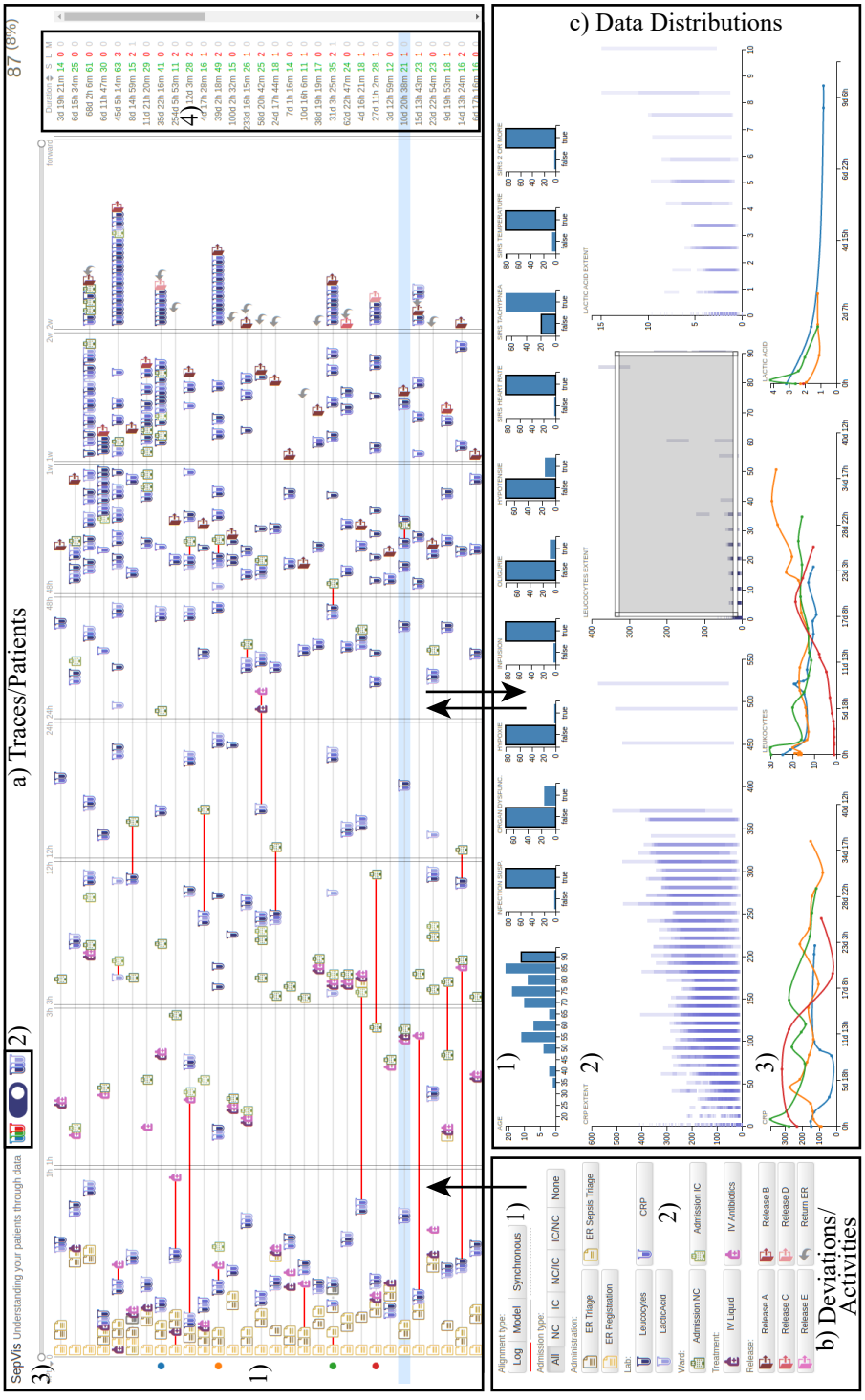


Figure 4.5: The components and interactions within SepVis: the *Pathways view* (a) shows all the pathways of patients, which are grouped by variants over time, the activity/deviations filtering panel b) and the clinical attributes distribution panel c). Details about performances are shown in a.4). Distributions of the clinical attributes for the current selection of patients are shown in c.1) by the orange bins. Distributions of the extents of the laboratory tests are shown in c.2). Single-patients trends, such as treatment values, are displayed in c.3).

underwent before entering a normal care unit, is integrated with the application. The user can use the *distributions panel* to create a cohort of patients and immediately recognize patterns in the *Pathways view*. In the following subsection, we illustrate these visualizations and their interaction in more detail.

4.5.1 **Pathways View**

This is the core view of SepVis and it consists of icons representing activities of the entire event log. Those activities are linked by lines to explicitly encode the temporal relation between two consecutive activities. The sequence of icons and lines within a row shapes a trace. Contrary to the current process-centric visualizations (see Figure 4.2), we distribute activities over time. The icons that belong to a particular trace are placed along the horizontal axis according to their accumulated duration since the starting activity of the trace, which is always considered to have an accumulated length of zero. Because we cope with data regarding a life-threatening condition where actions need to be taken in a few hours from the entrance of the patient in the hospital, the horizontal axis of this view represents time intervals with different scales. Following the guidelines [116], we opt for the following intervals: zero to 1 hour, 1 to 3 hours, 3 to 12 hours, 12 to 24 and then to 48 hours, 1 to 2 weeks and then onwards. Thus, this view comprises seven scales along the axis (one for each interval). All intervals are given a different horizontal space. The first intervals are wider to enable to focus on the events occurring in the first moments of patients hospitalization and the first taken actions.

The lines in this view also encode deviation information, which is provided by the process mining alignment algorithm. Typically, we can find two different types of deviations: log and model deviations (see also Section 2.3.1 and Section 4.4). The first deviation is encoded by coloring the line between two activities with a red color and increasing the thickness of the line. Because a line links two activities, we color lines red when the last activity occurs as a log move. The second deviation is depicted by using a dotted line(s) between the activity that makes a model move and the surrounding activities. This approach is susceptible to be cluttered because of icon overlaps. We solve this problem by shifting the icons when necessary, that is, when the link between two activities encodes meaningful information (i.e., deviations).

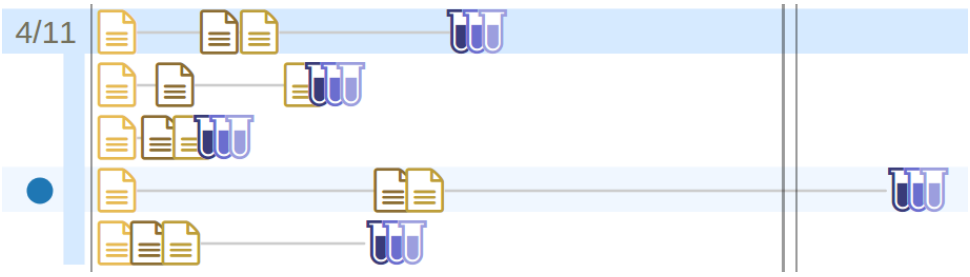


Figure 4.6: Example of an expanded variant that originally contains 11 traces in total and 4 of them match the current filter setting. A selected trace is indicated by the blue dot at the left of the trace.

The traces are grouped in variants, which represent the same sequence of activities for all the traces that they group. This grouping just happens when there is more than one trace of a given variant. The visualization of the variants is slightly different from the visualization of the individual traces. It collapses all the traces which belong to that variant and just shows the variant itself together with two numbers: one indicating the total number of traces for the variant, and another one showing the number of traces that match the current filtering setting. Moreover, the placement of the icons within a variant is based on the average accumulated duration, which is computed over all the traces that form a variant.

The variants can be sorted by different criteria by clicking on one of the fields on the header of the view. These criteria consist of several traces per variant, total (average) duration of the traces (variants) and delay between two selected activities.

The right side of this view displays a summary of how many deviations and normal moves a specific variant or trace contains. These figures are shown by using different colors as follows: synchronous moves are given in green, log moves in red and model moves in grey. This piece of information gives a quick summary of how well the trace fits the process model.

For each row in the visualization, four main interactions are available:

Clicking a variant. This interaction displays all the traces that belong to that group with a drop-down effect. The user can then explore each case. Figure 4.6 shows an example of an expanded variant. The traces that belong to the variant are placed beneath it and the variant is highlighted with a bluish color. A vertical thick line at the left of all the traces is used to highlight that they belong to the same variant and differentiate from the rest. When clicking on a variant, the data attributes of all the traces that it contains are reflected in the charts. This feature is explained in more detail in Section 4.5.4.

Hovering activities. Mousing over either icons or lines opens a tooltip, which shows information about the hovered element. When the hovered element is a line, it simply shows the elapsed time between the two linked activities. When it is an icon, it shows some basic information such as activity name, the time elapsed since the beginning of the trace and the time elapsed since the previous activity. Moreover, when the icon represents a more meaningful activity such as a lab test, the tooltip also includes information about the value of the hovered activity, or the minimum and maximum values for that test when hovering a variant. Finally, in case the hovered activity represents a log or model move, the tooltip also shows this information.

Hovering traces. Users can also mouse over traces that represent particular care flows, in which case the views depicted in Section 4.5.4 are updated showing the interesting values for the specific case. A more complete description of this interaction is given later.

Selecting activities. We enable the user to select singular activities. This interaction is described in Section 4.5.2.

Furthermore, the user can directly manipulate the view by the top slider (Figure 4.5(a.3)). The behavior of this slider is explained in Section 4.5.3.

On the header of this view, we can find different components. The upper right corner of the

screen displays the number of patients that match the current filter setting and the percentage of the number of patients. This information gives an idea of how representative the current population is. Also, on top of the view (Figure 4.5(a.2)) we can see the widget used to switch between the different color schemes that were depicted in Section 4.4.1. This component is presented as a switch. At both sides of the switch, we can find an example of icons that are colored with different color schemes. This indicates which color scheme we selected. Once the user selects one color scheme, the view updates to present the icons with the new color scheme.

4.5.2 **Activity filters and conformance checking**

In this section, we describe the behavior of the tool regarding filtering traces by activity or sequence of activities, and conformance information, that is, deviations in the process. This component can be seen in Figure 4.5(b).

Traces can be filtered out by the types of deviations they contain. Our tool provides four different options: all cases, only log moves, only model moves and only synchronous moves. This is useful to retrieve traces that entirely fit the process model or those that have specific deviations. This directly relates to task **T3**.

Also, traces can be filtered by activities by interacting with the buttons (see Figure 4.5(b.2)). This feature is provided to accomplish task **T2**. Clicking on a button in the activity box filters out the traces that do not contain such activity. Once this happens, the main view automatically reacts by decreasing the opacity of the other activities to highlight the selected activity. Furthermore, the information displayed at the right side of the main view automatically updates and shows the time elapsed until that activity occurs for each trace. In the case of variants, it shows the average elapsed time. Users can then sort traces by this new value.

This feature is extended by enabling users to select a second activity. In this case, the main view filters out the traces that do not contain both activities in the specified order, highlights both selected activities, creates lines between the two selected activities, and eliminates the other ones to present a view more focused on the selected activities. Also, similar to the case of selecting one activity, the main view shows the time elapsed between the first occurrence of the first selected activity and the first occurrence of the second selected activity. Traces can then be sorted by this new value.

The new information displayed in both situations can be used by the other components of the application to filter out traces and is explained in Section 4.5.3. This component also plays the role of legend in our application since it shows all the activities with their colors and their names.

An extra filtering mechanism is provided concerning activity filtering. This filter enables users to explore care pathways in which patients were admitted to NC, IC, NC and then IC, and IC and then NC units. In addition, the option to explore patients that went to neither NC nor IC is provided. These options address task **T1**.

4.5.3 **Temporal filters: Performances**

The slider located above the main view is aimed to filter out traces according to some time criterion. The criterion depends on the current context of the tool. Initially, this component just considers the total duration of the traces. Thus, users can shift both ends of the slider to set minimum and maximum thresholds for the duration of the traces in which they are interested. The slider follows the same time scales as the main view.

If an activity is selected as described in Section 4.5.2, the lower and higher values of the slider apply only to the first and last occurrence of the selected activities. Therefore, if users select only one activity, the slider will consider those traces in which the first occurrence of such activity happens after the lower value. If users select two activities, then both lower and higher thresholds will be considered to filter out those traces in which the first occurrence of the first activity happens after the lower value and before the higher value of the slider.

4.5.4 **Attribute filters: Cohort distributions**

The attributes panel is provided to look in more detail at patient's pathways from the clinical perspective. Among the available attributes, we can find the age of the patients and other clinical attributes of their conditions at the ER registration. A more complete description of the dataset is given in Section 4.3.

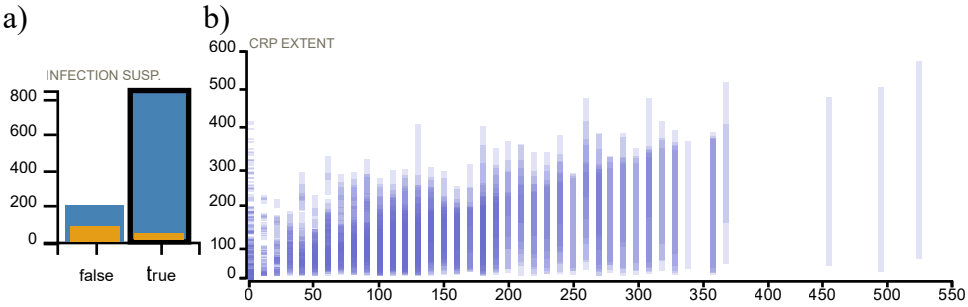


Figure 4.7: a) Bar chart showing three different pieces of data: the values for all the traces that match the current filters displayed as blue bars, the values for all the traces of the selected variants displayed as gold bars and the value for *infection suspected* attribute of the hovered trace displayed as a black border. b) Chart that aggregates patients with similar values for the CRP tests.

Clinical attributes are shown as individual, interactive bar charts. Interacting with one of the bar charts triggers an update from the rest of the charts and views of the tool. Also, these charts are linked with the main view when a variant is selected and a trace is hovered. When this happens, two outcomes are possible (see Figure 4.7(a)):

- If a variant was selected, the data from all the traces that belong to it is displayed by showing overlapping bars for each bar chart. These bars are horizontally shrunken and positioned on top of the corresponding bars. Moreover, the color of these bars is different to easily spot the difference between the whole data and the selected traces.

- If a trace was hovered, then the bars with the values of the data of that specific trace are highlighted by adding a black stroke. This way we can explore the attributes for particular traces.

The data view also includes charts to present the values for different laboratory tests that are available in the dataset. Because there is more than one value per test, we opt for using two charts per type of test: one chart that shows a summary of the values for all the traces currently in the main view and another one which displays all the values of that test for traces selected by the user.

The first type of chart (see Figure 4.7(b)) provides a representative picture of the range of values for a specific laboratory test. In our approach, we use a histogram in which we group patients by the difference between the global maximum and minimum values for the laboratory test. The horizontal axis is divided into slots of the same size (10 units in the example), and each data item (patient) is horizontally assigned to the closest slot based on the difference of the extreme values. Each patient is depicted by a bar. The height of a bar encodes the variation of the value for the laboratory test. Moreover, each bar is assigned a semitransparent filling color. This representation enables to detect where the trend of the distribution of values is and the existence of outliers at the same time while keeping the whole picture of the data. Specific parts of the data can be selected by brushing over the chart, making the rest of the tool reacts to show just the selected patients.

In the bottom part of SepVis (see Figure 4.5), we can find the charts which display all the values of a particular test for the selected traces. This type of chart is aimed to give a better understanding of the evolution of the patient over time. Therefore, the horizontal axis of this type of chart encodes time and the vertical axis encodes the values of the test. This type of chart only offers the interaction to see the value of specific points by mouse hovering them.

These charts, and the attributes that they depict, are interesting for domain experts that are aware of guidelines since normal/abnormal values might indicate some decisions that may need to be taken. With the interactions described in this section, we enable users to explore the distribution of these attributes in a proper way.

4.5.5 **Implementation**

SepVis is implemented as a web application and it is entirely programmed in D3.js v4 [18]. We use Bootstrap v3 for a base layout, in combination with Icon Moon icons. Finally, we make use of several JavaScript libraries: jQuery v1.12, underscore v1.8 and Crossfilter.js v1.3. Finally, we use color palettes from D3.js that were taken from ColorBrewer.

Regarding process mining technologies, we use the alignment technique described in [172] which is available as a plugin in ProM [177].

4.6 **Use Cases and Results**

We demonstrate the usefulness of SepVis by some use cases with questions extracted from conversations with doctors involved in the project [99] and the guidelines provided by the Surviving Sepsis Campaign [67] released in 2012.

4.6.1 **Clinical pathways in NC and IC units**

In an emergency department for suspected sepsis patients, one of the first questions that the clinical personnel tries to answer concerns clinical pathways across the treatment units.

For instance, it is interesting to know the flow of patients that are only admitted to the NC ward, only to the IC unit, or first to the normal care unit and, only then, to the intensive care unit. The last category is considered of great interest by clinical personnel because it hints to patients whose conditions have worsened after being admitted. In these situations, fast decisions need to be taken. Given the high interest in these particular activities, we provide a dedicated button bar for selecting these particular groups: patients that entered respectively only in the Normal Care unit, Intensive Care unit, Normal Care and then after Intensive Care unit, or the other way around. A special group is also included for patients who entered neither Normal Care nor Intensive Care. This group of patients might reflect those who went to the emergency room and did not need to be monitored.

Our dataset has 67% of patients that went only to the NC unit; 23% of them did not enter in *any* of the two units during their stay at the ER. Selecting the latter option, the users can notice that, although these should be the cases with the shortest duration, some patients activities have been recorded even after one day. In total, the number of patients that present deviations (model moves and log moves) was around 48% of the total cases. Of this group, eleven patients are likely to match the diagnostic criteria for a sepsis condition. The diagnosis of an inflammatory state is typically performed by assessing the manifestations of the Systemic Inflammatory Response Syndrome (SIRS), such as alterations of body temperature or increase of heart rate and white blood cells. Therefore, we can look for patients with a clear need of specific care by filtering with SIRS-2 criteria (patients matching two or more SIRS criteria), suspected infection and positive organ dysfunction, the major indicator of a sepsis condition [35]. The user gets a list of patients (1% of the total patients). For some reason, perhaps mistakes in recording patients activities or other decisions, these patients have not been treated in the specific care units. Switching back to *all* patients, we get around the 7% of the cases whose treatments are lasting more days because of follow-up and monitoring tests.

4.6.2 **Performances and delays**

Guidelines for life-threatening conditions provide clear recommendations regarding treatment administration time. Early identification of septic patients is crucial to prevent delays

in management and treatment. Many studies focus on the impact of time on antibiotics administration in respect to triage screening tools and protocols [48, 122, 166]. Current guidelines in the management of sepsis patients [35, 67] state that the distribution of antibiotics to patients in sepsis conditions needs to take place within the first hour of a sepsis triage. Due to lack of resources or delays, this might not happen, leading to a serious risk to the patients, which has to be avoided. In SepVis, the users can get insight into these circumstances. Firstly, they select the 'suspected' sepsis patients according to the clinical attributes at disposal. Next, the user chooses the *ER-Sepsis Triage* event as the first option and *antibiotics* as the second from the activity list. Thus, we get the resulting patients in the *pathways view* and the elapsed time between two activities. The user can observe that many patients received antibiotics between 1 and 3 hours and after 3 hours. Most likely these patients were not considered to be in a critical condition, therefore the administration was delayed or simply recorded after some time. In this view, we can also see some deviations in the clinical pathway. This is because of the proposed protocol used in SepVis, which states that *intravenous antibiotics* is a treatment followed or proceeded by also *intravenous liquids* administration. In some cases, it seems that this administration of liquids was not performed or recorded.

Another interesting case is visible by considering the *3-hours bundle* of [67]. In order to simplify and improve the care of patients with severe sepsis, the *Surviving Sepsis Campaign* released bundles, which are a selected set of elements to be considered and executed to aim at the best outcome. In the 3-hours bundle, patients with hypo-tension or lactate values below 4 mmol/L need to be treated with *intravenous fluids* to expand their circulating volume. The user can check the patients matching the hypo-tension attribute and lactate values above the threshold. In our scenario, the dataset comprises four patients that got IV fluids after 3 hours. In combination with EHRs, this kind of result can be explored in more depth. Unfortunately, the accuracy of the clinical attributes at our disposal is not sufficient to conclude the outcome of these suspected delayed interventions.

4.6.3 Deviations and clinical attributes

As we described in the Background section, in process events, two different kinds of deviations can be detected by conformance checking: log-moves and model-moves.

With SepVis, we enable the users to see when and where these deviations occur. From the clinical perspective, a log move can be described as a decision that does not match the expected behavior. A model-move instead, might indicate forgotten actions, missed decisions, lack of resources and such.

Thus, we enable the user to filter out patients that matched entirely the proposed model. At this point, deviating cases can be analyzed over time and by clinical attribute distributions. Once the deviating cases are available, the user might be interested to see single patients' trends in test values over their stay in the hospital. The user can select a few patients from the deviating case list. Then a click on a test activity such as *CRP*, *leukocytes* or *lactic-acid* levels, triggers the update of the respective charts showing the fluctuations over time.

4.7 **Conclusion**

In this work, we have proposed a visualization approach that enables the analysis of process-centric information of an event-log in combination with clinical longitudinal data.

We have shown the usage of the current implementation in accordance with standard clinical tasks and relevant insights in respect to current sepsis management guidelines. Our design comprises a single interface, which is compact and more intuitive in comparison to current solutions in the process-mining domain. It provides the necessary components to interact with cases and at the same time analyze their clinical pathways filtering by time, performances and clinical data.

The core components of SepVis allow the exploration of an event log by interacting with its main elements: cases, activities, and attributes over time. While we demonstrated SepVis on an example dataset for sepsis, our approach is generic and can be used with other event logs. The current limitation is given by the fact that our visual interface has been built for an event-log with a low number of activities (16) whereas processes may comprise hundreds of different activities. At present, SepVis exclusively supports a single-cohort investigation per time.

5

Interactive Correction of Deep Learning Predictions in Sleep Staging

The usage of deep learning models for tagging input data has increased over the past years because of their accuracy and high performance. A successful application is to score sleep stages. In this scenario, models are trained to predict the sleep stages of individuals. Although their predictive accuracy is high, there are still misclassifications that prevent doctors from properly diagnosing sleep-related disorders. This chapter presents a system that allows users to explore the output of deep learning models in a real-life scenario to spot and analyze faulty predictions. These can be corrected by users to generate a sequence of sleep stages to be examined by doctors. Our approach addresses a real-life scenario with an absence of ground truth. It differs from others in that our goal is not to improve the model itself but to correct the predictions it provides. We demonstrate that our approach is effective in identifying faulty predictions and helping users to fix them in the proposed use case.

The contents of this chapter have previously appeared in **Garcia Caballero, H. S., Westenberg, M. A., Gebre, B., and van Wijk, J. J.** V-awake: A visual analytics approach for correcting sleep predictions from deep learning models. vol. 38, pp. 1–12 [52].

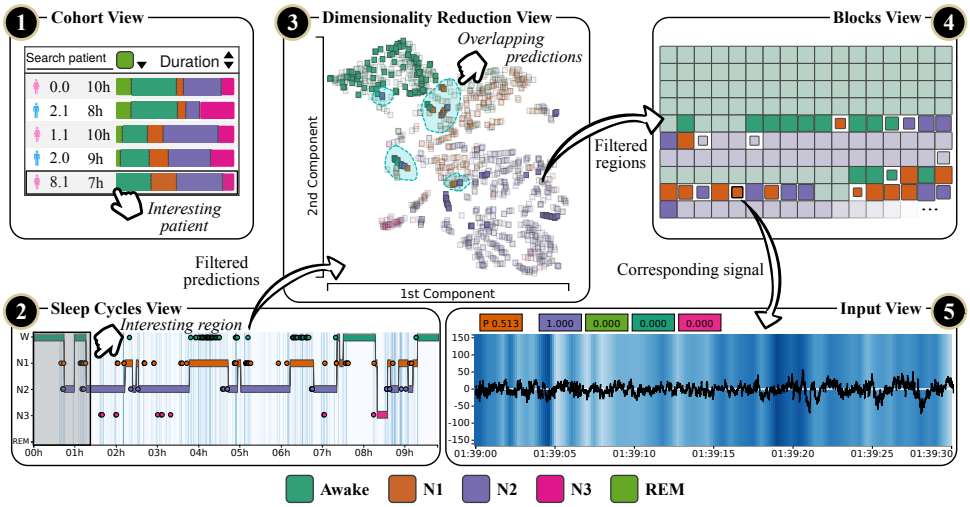


Figure 5.1: Depiction of the main components of V-Awake. First, a patient is selected (1) and the predictions from the deep learning model are displayed in 2, 3 and 4. Next, some of the predictions are selected (2) and the data in the dimensionality reduction plot is highlighted (3). Some regions in the scatter plot are selected and the corresponding predictions are marked in the blocks view (4). Finally, selecting a prediction block makes the input view display the corresponding input (5), which can be analyzed to determine if the prediction is correct.

5.1 Introduction

The usage of deep learning (DL) has notably increased in the past years due to its effectiveness to solve problems of different nature. The applications of DL models are many and can be found in a wide range of contexts. For example, they have proved to be effective for many image-analysis tasks: object recognition [196], image captioning [24, 44, 185], image segmentation [118] or image classification [83], to name a few. Another successful domain is the medical field, wherein DL models were developed to help practitioners with their daily tasks: lung nodules detection and classification [68], or nuclei detection and classification [5, 33, 150].

One important field within the medical domain is the study of sleep. In this context, individuals are subject to polysomnography (PSG) tests when they are believed to be suffering from sleep disorders. The PSG involves measuring brain signals, which are recorded by electroencephalography (EEG) and analyzed afterward by an expert. This expert is in charge of scoring the PSG by tagging pieces of the whole recording as sleep stages. This manual approach is time-consuming and labor-intensive [15], making it hard to apply at a large scale. To overcome this limitation, a lot of research has been performed to automate sleep scoring tasks, for example, by using DL models [15, 90, 163, 167, 199]. The automation of the scoring process has obvious benefits regarding time and effort. However, it also brings drawbacks in terms of reliability and accuracy of results.

In the medical field, it is even more important than in other fields that models produce correct outputs to solve other complex, human-dependent tasks (e.g., diagnosis). Although models provide certainty for the predictions, it does not depict actual validity that can be used to ensure any correctness. In addition, in real-life scenarios there is a lack of ground truth, making it nearly impossible to ascertain whether a prediction is correct or not. As a result, a reviewing process is necessary to ensure a certain degree of correctness. This reviewing process eliminates all the benefits of automation because it requires an inspection of the whole output space.

In state-of-the-art work, many tools support the development of DL models [74, 75, 92, 93, 107, 126, 161, 192]. Generally, they enable users to see whether a model performs correctly in terms of predictive accuracy, for example, by finding superfluous layers or deficiencies in the training data. Nevertheless, all these tools are applied in a development stage to improve a model. In contrast, our work focuses on a real-life scenario in which we have an imperfect model and no ground truth any longer. Therefore, we aid users in an exploratory process to find the potentially misclassified predictions. Our approach does not aim to discover the cause of the misclassification.

To tackle the difficult task of finding misclassifications in a real-life scenario, we present *V-Awake*, a visual analytics approach that aids users to find, store, analyze and correct faulty predictions from DL models. Our contributions are: **1)** We present the first visual analytics system for deep-learning based sleep staging, and **2)** We apply visualization in the absence of ground truth, i.e., real-life data, to accelerate detection of misclassification in deep-learning based sleep staging.

We conducted our research with a sleep scoring model trained on raw, single EEG channel data [163]. We demonstrate the usability of our approach in a concrete, real-life use case together with two somnologists. We discuss the limitations of our approach, how it can be generalized to other domains that use DL models, and we give directions for future research.

5.2 **Medical Background**

The study of sleep is an important area in medical research. It can reveal disorders, such as apnea, narcolepsy, parasomnia or hypersomnia, which can also relate to other types of medical conditions such as psychiatric disorders, neurodegenerative diseases [193] or cardiovascular disorders [153]. Therefore, having a good understanding of sleep is crucial to provide better diagnoses of some diseases.

The current procedure to study sleep patterns in clinical settings consists of several steps. First, a PSG is performed to record brain signals, eye, chin and leg movements, blood oxygen level, heart rate and breathing of the patient. After the recordings have been obtained, a PSG technologist determines sleep stages by applying rules defined in one of the major sleep scoring guidelines such as the Rechtschaffen and Kales [134] or the American Academy of Sleep Medicine (AASM) [119]. The stages are usually tagged in 30 seconds segments of the PSG, which are called *epochs*. Subsequently, a sleep doctor uses this information to make a diagnosis.

The main drawback of such an approach is the amount of time needed to score sleep stages. For instance, a technologist may spend over one hour to score an 8-hour PSG [15] due to the labor-intensive nature of the scoring process that involves the analysis of several indicators. In Figure 5.2 five examples of EEG signals are shown depicting the five sleep stages described in the AASM manual [119]. Stages *N1*, *N2* and *N3* represent the non-rem stages, and *REM* indicates the rapid-eye-movement phase of the sleep. Each stage is characterized for having different morphological characteristics going from lighter to deeper sleep respectively. As can be seen, the distinction between different sleep stage patterns can already be hard when analyzing the signals in isolation. In addition, technologists have many other parameters to be considered (e.g., movement sensors, oxygen level in blood, breathing rhythm, etc.), increasing the complexity even further.

Our approach aims to aid experts that score PSGs (i.e., technologists) at correcting the output of DL models for sleep stage scoring.

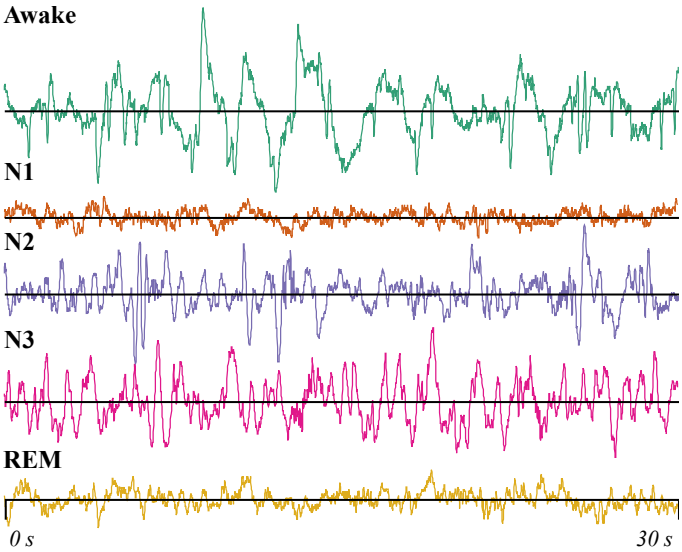


Figure 5.2: Five 30-second epochs depicting the five different sleep stages described by the AASM. Stages from top to bottom represent respectively lighter and deeper sleep. Stages *N1* to *N3* depict the *non-rem* stages, while *REM* stage depicts the rapid-eye-movement phase of sleep. Signals are recordings from *Fpz-Cz* derivation [81].

5.3 **Related Work**

Most of the visualization approaches available in the literature focus on the *development* of a DL model. Generally, the goal is to find issues in training/validation data, or architectural deficiencies like superfluous layers, non-suitable activation functions, etc., that can be used to modify the initial model to create an improved version of it. In this section, we provide an overview of techniques that deal with understanding models, paying special attention to DL models. Our proposed approach distinguishes itself from previous work in that we work

with a real context that lacks ground truth. This scenario has not yet been considered in the literature [65].

5.3.1 **Performance Analysis**

Work has been done on performance analysis of predictive models from a general perspective, focusing on the visual exploration of several performance indicators. ModelTracker [7] and Squares [136] are two systems that intend to provide insights into the performance of classifiers. Although they share a common goal, their approaches differ. The former system presents both training and test data, enabling users to label data as positive or negative, tag groups and link them through iterations of the model. Squares, for its part, strives to analyze the performance of multiclass classifiers. To this end, it visually presents the results of the validation data in a parallel coordinate plot fashion in which each column represents a class. This enables a comparison of the performance per class. The two systems differ from ours in that they address scenarios with ground truth available, and their goal is to analyze the performance of models.

5.3.2 **Neural Network Analysis**

Much work has been done on understanding convolutional neural networks (CNN) [74, 93, 126] and recurrent neural networks (RNN) [75, 107, 160, 161].

Their main goal is to provide insight into what networks learn. To this end, some techniques make use of 2D projections in combination with labeled data to find what Liu et al. [93] call *pure* and *impure* clusters. These cluster types indicate good or bad splitting of the data respectively. Therefore, they can be used to investigate how the model performs. ActiVis [74] uses 2D projections of the activations of several layers to determine whether the model learned how to properly split the input data into classes or not. Similarly, DeepEyes [126] utilizes projections to identify stable layers. This system aims at helping during the training process, whereas ActiVis focuses its analysis in a post-training step. Interestingly, Rauber et al. [133] conducted several experiments on different datasets to demonstrate the usability of projections to evaluate how well the models learned to split the data. All these systems utilize 2D projections in conjunction with ground truth, that is, labeled data, whereas our approach is meant to use those projections solely due to the absence of ground truth. Therefore, we assume the network can produce good splittings which can be used for further analysis.

5.3.3 **Model Interpretation**

In some cases, experts use models to perform complex tasks like segmentation of anatomical structures or risk monitoring. In this context, models provide predictions or alarms based on given data. Inspection of model output is needed to ensure quality and be aware of possible misbehaviors. The work of Raidou et al. [131] presents a system that aims to help clinicians to understand segmentation models. It enables the exploration of errors in the segmentation

to find patterns that can help evaluate the reliability of the model. The previous work uses labeled data to guide the exploration. In other scenarios, the only available data is the steps performed by the model. The work of Scheepens et al. [142] aims at visualizing the rationale of a reasoning engine that is fed with possibly unreliable sources. Due to the nature of unreliability, experts require a support system to discard possible false alarms.

Both examples reflect an actual necessity to support users when using a model in a real-life scenario. Nevertheless, the concepts introduced in these works cannot be translated directly to the sleep staging problem nor DL models.

5.3.4 Explanation Techniques

Explanation techniques are used in complex systems to provide a better understanding of DL models. This area has recently drawn attention due to the necessity for experts to explain how models work. For providing explanations on CNNs, a great amount of work has been done [97, 148, 156, 197]. We focus on the techniques that are most closely related to our approach.

Fong et al. [45] compute a perturbation mask that indicates ranges of the input space that were salient for the model when making a prediction. Other studies address the same problem with different approaches. For example, Grad-CAM [144] tries to find salient regions in the input space using gradients applied to the last convolutional layer of a CNN. It generalizes an earlier work that introduced a method to compute the so-called Class Activation Maps (CAM) [200], which also depicts an approach to find saliency regions. The main drawback of CAM is that it is restricted to CNNs that do not include fully-connected layers. Another approach was introduced by Zeiler et al. [196] to find salient regions by occluding parts of the input and attaching a *deconvolution* net to the model we want to analyze. All these techniques share a common goal, although their methods differ.

Regarding explanation techniques for RNNs, Van der Westhuizen et al. [176] apply existing saliency methods like the ones described previously to temporal inputs (electrocardiogram). They found that deletion masks provided the best results and saliency regions matched medical concepts like types of waves that are used to recognize patterns. Regarding temporal inputs and hybrid models that combine convolutional and recurrent layers, recent work [51] analyzes saliency approaches and shows that they do not suffice to provide good explanations. Through visualization, they demonstrate that more research is needed to better understand how this type of model works with temporal input data.

5.4 Problem Definition

We define the problem of correcting predictions in a neural network model using tasks that depict the main goals in our system. Our goal is not to improve a given model in terms of predictive accuracy. Rather, it is to find incorrect predictions in environments in which ground truth is missing to enable users to correct and indicate what the prediction truly is.

In this section, we firstly introduce a description of the model and data that we use in our approach. Next, we define a set of tasks and give a brief description of them.

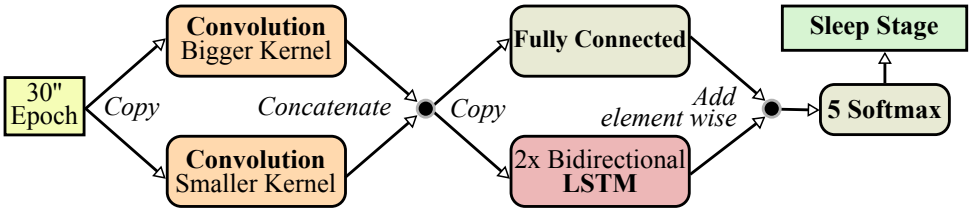


Figure 5.3: Deep learning model for sleep stage scoring [163]. It comprises two convolutional branches with different kernel sizes, a shortcut connection and two bidirectional LSTM layers.

5.4.1 Model Description and Dataset

To introduce our approach, we use a DL model [163] that scores sleep data. A graphical, high-level description is shown in Figure 5.3. It has two convolution branches with different kernel sizes: a smaller one to capture temporal information (i.e., EEG patterns) and a larger one to capture frequency information (i.e., frequency components). The derived features are then concatenated and fed to the recurrent part of the model, which is formed by two bidirectional long short-term memory (LSTM) layers. A residual learning approach is used, which is stated by the fully connected layer parallel to the bidirectional LSTMs, to keep track of the features extracted in the convolution step. Finally, all these activations are added up and fed to a fully connected layer with a 5-softmax activation function that serves to normalize the output into a probability distribution of five classes. The model is trained in two steps using data from a sleep study [81] available on PhysioNet [56]. In the first step, the representation learning (i.e., convolutional layers) is done. Next, a residual learning approach is used to train the two LSTM layers as well as the shortcut connection (i.e., fully connected layer in Figure 5.3). Once the model is trained, it can be used without the necessity to retrain. In this model, convolutional layers act as feature extractors directly from the raw input signal, while LSTM layers learn transition rules between sleep stages. The model achieves an accuracy of 82.0% [163].

The data used to train the model represents the sleep recordings of 20 patients (from subject *SC4001E0* to subject *SC4192E0*) over two nights. A depiction is shown in Figure 5.4. For each patient j and session k, a signal $f_{j,k}(t)$ is measured, where $t \in \mathbb{Z}$ indicates a point in time in seconds. The signal is sampled at a frequency of $100Hz$, resulting in 100 measurements per second. $E_{i,j,k} = [f_{j,k}(30i), f_{j,k}(30i+1), \cdots, f_{j,k}(30i+29)]$ represents the i-th epoch, which is a 30-value vector for patient j and session k. Each epoch $E_{i,j,k}$ is 30 seconds long, accumulating a total of 3000 values. Finally, $C_{i,j,k}$ indicates the corresponding classification for the i-th epoch of patient j in session k.

The signal f is gathered from sensors placed on the head of the patient during sleep. On average, there are 1075 epochs per patient and session, resulting in 1075 predictions over approximately 9 hours of sleep and above 3 million points. Examples of signal f for each

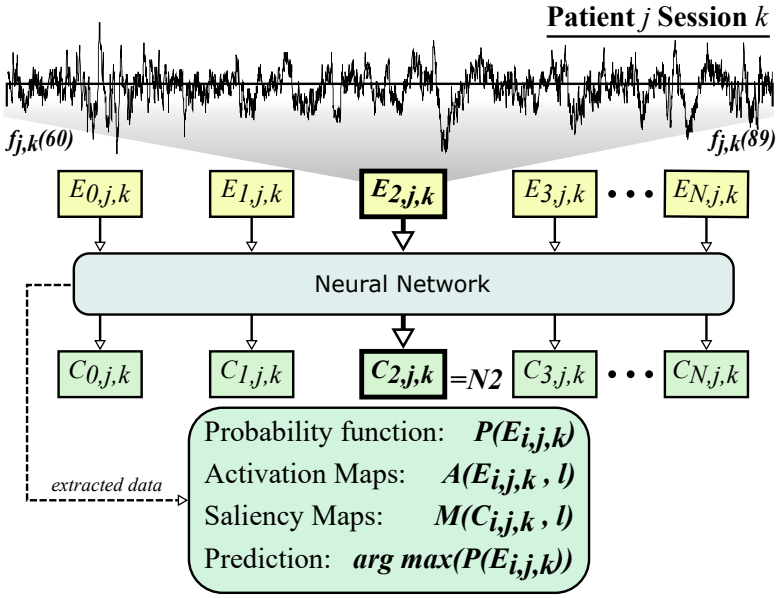


Figure 5.4: Illustration of the data used in our approach for patient j and session k. An example of a signal is given for epoch $E_{2,j,k}$ which is classified as stage *N2* after being run through the model.

sleep stage are shown in Figure 5.2. Besides all this data, we retrieve the following information from the model:

Probabilities for each possible class are provided by DL models in classification. Thus, a function $P(E_{i,j,k})$ provides a vector $P_{E_{i,j,k}}$ of probabilities where $P^c_{E_{i,j,k}} \in [0,1]$ depicts the likelihood of epoch $E_{i,j,k}$ being classified as class c. Analogous, $P^c_{E_{i,j,k}} = P(C_{i,j,k})$ where c is the class predicted for epoch $E_{i,j,k}$, that is, the class with the highest probability.

Activation Maps, also named feature maps, depict the output produced by a certain layer l in a DL model after applying an internal function. This function depends on the type of layer. For instance, convolutional layers apply *convolution* over the input data to derive new features, i.e., produce an output. As evident, these features are used to determine the classification of unseen, new input data. Therefore, they can be used to discover similarities in predictions. The function $A(E_{i,j,k}, l)$ retrieves the activation map for epoch $E_{i,j,k}$ and layer l, containing a variable number of activations $u_{i,j,k,l}$.

Saliency Maps describe how important the attributes of the input data are for a layer of the model to predict that input as a particular class. The function $M(C_{i,j,k}, l)$ provides the saliency map for the epoch corresponding to classification $C_{i,j,k}$ and layer l. In our case, each saliency map contains a constant number of values $v^m_{i,j,k,l} \in [0,1]$ with $m \in [0, \cdots, 2999]$, that indicates how important the m-th value is for layer l to classify $E_{i,j,k}$ as $C_{i,j,k}$. Our approach uses Grad-CAM [144] to compute saliency maps. The dimension of the map this method gives depends on the output size of the

layer. However, to keep a constant size, the values of the output are rescaled with linear interpolation to match the size of the input instances, i.e., 3000 values.

5.4.2 **Tasks**

Based on multiple interviews with two somnologists and four DL experts, we define a set of tasks that are considered relevant for the analysis of DL predictions for sleep scoring:

T1 **Fix incorrect predictions**. Incorrect predictions are a serious problem. Finding them is not a trivial task when there is a lack of ground truth. Hence, users, independently of their expertise, should be enabled to explore the data in such a way that they can find potentially incorrect predictions and repair them by indicating the actual class.

T2 **Understand why the model made a prediction**. Once potentially incorrect predictions are found, it is necessary to understand why the model made such a prediction. This helps users to understand whether the prediction is correct or not.

T3 **Re-tag predictions with basic support**. The system must allow users to re-tag selected predictions. Hints must be provided to users to help them make a decision.

We designed a workflow to support the defined tasks (see Figure 5.5). Users can perform actions in whichever order they decide.

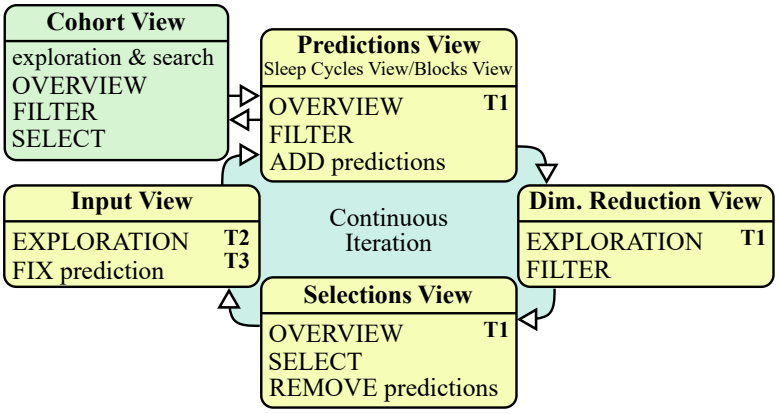


Figure 5.5: Depiction of our workflow and the mapping to the views of our design. Arrows depict a common way of interaction, although other routes are possible. Upper case words summarize the most important actions performed in each view. The tasks that each view performs are also shown in the diagram.

5.5 **V-Awake**

In this section, we introduce the main components of our approach (see Figure 5.6 for an overview). Although it was designed for sleep experts, the components are generic enough to be used by experts with different backgrounds (**T1**).

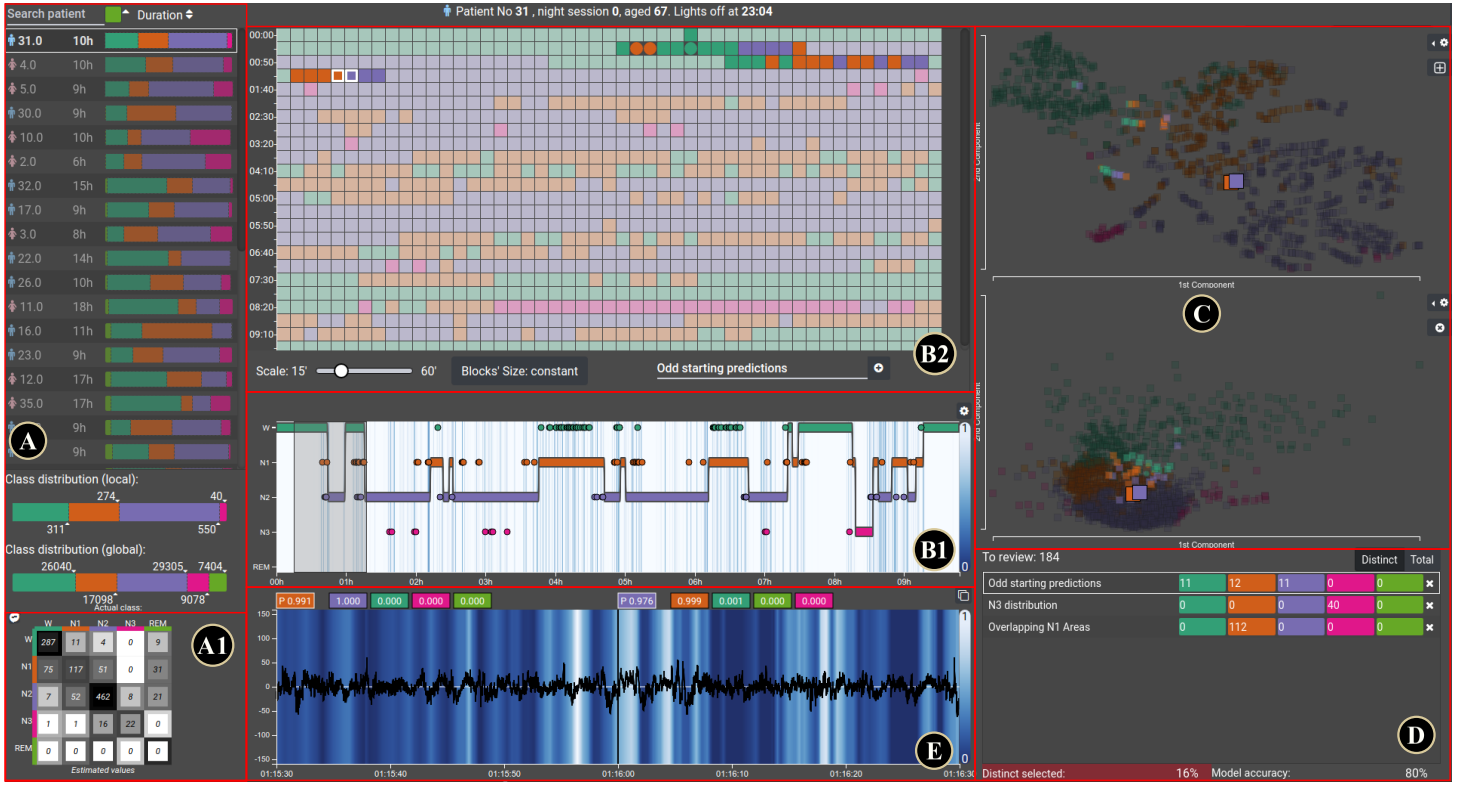


Figure 5.6: The user interface of V-Awake. Numbers depict the views and components of our approach: cohort view (A) and confusion matrix (A1), predictions view with sleep cycles view (B1) and blocks view (B2), dimensionality reduction view (C), selections view (D) and input view (E).

5.5.1 **Cohort View**

The primary goal of the cohort view (Figure 5.6A) is to provide a summary of the data for each patient j and session k. The summary displays information regarding the gender of the patient, identifier and length of the recording session, and distribution of the predictions $C_{i,j,k}$. This provides an overview that can be examined to spot interesting cases for experts.

The summary of the predictions plays an important role because it might show particularities that are of interest (e.g., absence of predictions of a particular class, or an abnormal class distribution). It is depicted with a horizontal stacked bar chart per patient. The width of each bar encodes the ratio of predictions of a class relative to the total number of predictions.

The ultimate goal from a usability perspective is that the user selects a case of interest. To facilitate it, *search*, *sort* and *layout* features are provided. Regarding the latter feature, users can select a stage of interest to be placed as the first element in the stacked bar chart to make comparisons between different subjects.

Below the panel that displays all the patients, two stacked bar charts are shown in the same fashion. The top one shows the distribution of predictions for the patient selected in the cohort view. The bottom one shows the aggregated distribution for all the patients. The location of both charts facilitates comparison of the local (i.e., selected patient) and global (i.e., all patients) distributions. Moreover, the summary view (Figure 5.6A1) provides information regarding the results of the model for the validation dataset. A confusion matrix is shown, which can be used by users to determine the cases that the model fails more often. When a patient is selected, estimated values are provided. These values are computed by interpolating the global values from the validation set of the model.

5.5.2 **Predictions View**

Predictions are the core items in our work. Our approach provides two different ways to directly interact with them. They are discussed in the following subsections.

Sleep Cycles View

The sleep cycles view is presented in Figure 5.6B1 and provides an overview of the whole sleep session in a familiar manner to the expert. It emphasizes the transitions between sleep stages. This *piano roll* representation enables users to spot interesting patterns quickly. The view is based on a time series chart where x and y axes denote time relative to the beginning of the recording and sleep stage, respectively. Colors are used to encode stages. The background displays the fluctuations in the certainty of predictions (i.e., probabilities $P(C_{i,j,k})$), without interfering with the core part of the view. Fluctuations can be utilized by the expert to spot regions in which the model was less certain and therefore prone to misclassifications.

To prevent visual clutter in the global trend, a preprocessing step is applied to the data to extract possible outliers. It consists of extracting consecutive prediction sequences that belong to the same class and contain fewer predictions than a set threshold. The threshold can be

adjusted by the user. For example, users can set a lower value to extract outliers that form very quick transitions. This helps to find cases in which the model rapidly changes stages, potentially indicating that there are misclassifications. The visual encoding in the sleep cycles view highlights transitions that should not occur in a normal context. In a normal sleep pattern, transitions should happen in a specified order. For instance, it is not possible to immediately move from *awake* to *REM*. Outliers are visually represented as dots visually disconnected from the main trend, which is represented as a *piano roll*. This particular visualization supports the task of spotting faulty predictions (**T1**).

The view relies on brushing to focus on a specific area of the prediction space. The other views are updated accordingly restricting further actions to the selected area. Furthermore, when an action is performed in other components, the sleep cycles view updates accordingly by visually de-emphasizing corresponding predictions.

Blocks View

Similar to the sleep cycles view, the blocks view (Figure 5.6B2) depicts an overview of all the predictions generated for a selected patient and session, which are represented as blocks and are placed sequentially from the top-left corner to the bottom-right corner. The major difference with the previous is the visual encoding and the interactions. While the sleep cycles view emphasizes transitions between stages, this focuses on the sequentiality of the predictions. The blocks view serves two main purposes:

1. **Give an overview of the predictions in constant time intervals**. Intervals directly connect with medical concepts (e.g., *N1* should last up to 7 minutes). To provide more flexibility, time intervals are adjustable by users, allowing the exploration of the predictions from different time perspectives.

2. **Highlight the predictions that are under consideration**. The location of this view is ideal for depicting the predictions that are filtered out from other components. This allows users to be aware of the time position of the predictions that are selected after performing brushing in other views (see Figure 5.6). Moreover, users can directly add or remove elements by clicking them. This provides fine-grained control over the elements that are currently selected. This control is useful to incorporate or exclude predictions that are located nearby in time and that the expert considers to be interesting for further analysis.

The size of the blocks can encode extra information such as the probability and the entropy of a prediction. The latter is defined as:

$$-\sum_{c=0}^{n} P^{c}_{E_{i,j,k}} \cdot \log_n P^{c}_{E_{i,j,k}},$$

where n is the number of classes. This encoding helps to visually identify predictions that deviate from others in terms of probability. The probability metric emphasizes high prob-

ability predictions, while entropy emphasizes extreme cases in which the probability values are very similar.

5.5.3 **Dimensionality Reduction View**

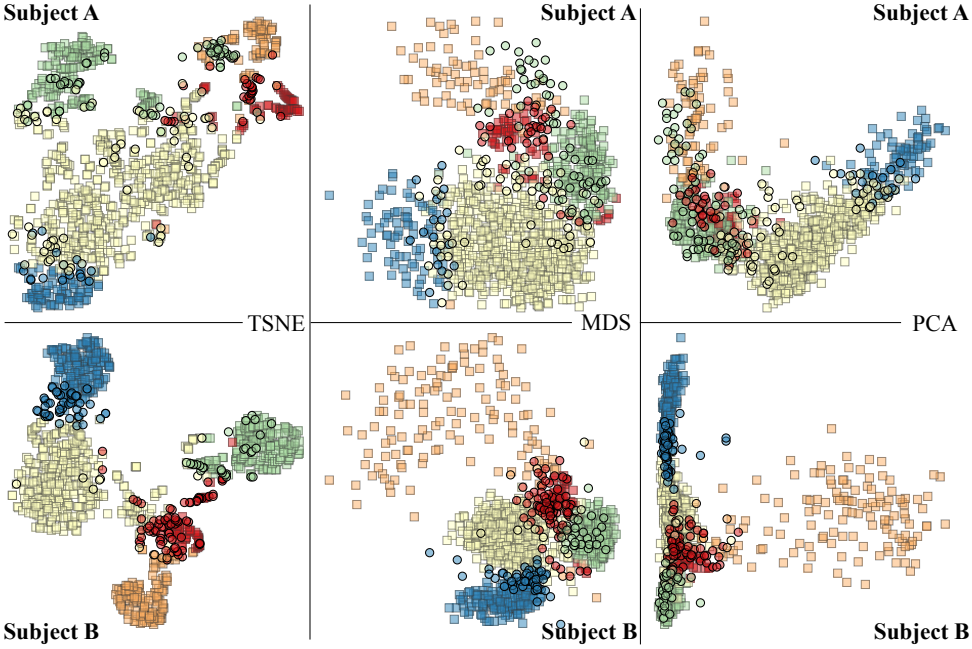


Figure 5.7: Comparison of tSNE, MDS and PCA for two subjects in a setting with ground truth available. Each square represents a prediction, the color depicts the predicted sleep stage. Circles with a black border represent predictions from the model that do not match the label assigned as ground truth. tSNE works, in general, better than other methods to identify borderline cases.

The dimensionality reduction view (Figure 5.6C) depicts a scatter plot with a dimensionality reduction computed over all the activations of a layer on the predictions of a given patient and session. It is used to find incorrect predictions by identifying visual overlaps (i.e., *impure clusters*). The data displayed in this view is defined as:

$$\sigma([A(E_{0,j,k}, l), A(E_{1,j,k}, l), \cdots, A(E_{N,j,k}, l)], n_c),$$

where σ represents a dimensionality reduction function and n_c is the number of principal components that σ provides. The rationale for the election of a layer is that each layer in a DL model learns different features from data. The layers close to the output are believed to effectively separate the features linearly [40].

We considered three dimensionality reduction functions: Principal Component Analysis (**PCA**) [66, 73, 123], Multidimensional Scaling (**MDS**) [84] and t-distributed Stochastic

Neighbor Embedding (**tSNE**) [96]. Figure 5.7 shows a comparison of them for two different subjects in a context with ground truth. The same model as in our approach is used to compute the projections. Circles with black borders depict misclassified predictions. As can be seen, PCA tends to group cases belonging to the same class and does not discern misclassified cases, hence we discarded this method. On the other hand, MDS and tSNE appear to better divide the space so that boundary cases for incorrect predictions stand out more. tSNE is the default option in this view. The design of this view addresses task **T1** since the location of the predictions in the plot might provide useful information.

Generally, dimensionality reduction techniques use heuristics to find the most optimal solution. They involve randomization, resulting in a different output every time the method is executed. Even though this randomization has benefits, it can worsen the exploratory process since the user would not get the same result in two different executions. To prevent this from happening, we decided to set the seed to a fixed value. Nevertheless, we provide users with options to set a random or a different value for the seed if desired.

We heavily rely on linking and brushing to help users spot suspicious elements, with the previously introduced components all coupled. Multiple dimensionality reduction plots can be visualized at the same time. They all are coupled, enabling an exploratory process in which we could compare different aspects:

1. **Different layers**. Users might be interested in exploring the dimensionality reduction output for activations of different layers. This is useful, for example, when the model has a hybrid architecture. Exploring the activations of convolution and recurrent layers side by side can lead to interesting findings. By default, the previous-to-last layer is selected because it has been shown that it performs the best for these methods.

2. **Number of principal components**. In most cases, the use of two components enables a fairly sufficient exploration. However, when this is applied, the sequential nature of the predictions is lost. To tackle this, in our approach it is possible to switch from 2 to 1 principal component, used for the vertical axis, while the horizontal axis is used for the sequence number of the epoch.

   When this occurs, the plot adapts the axes to either show both components or the only main component together with the sequence number of the prediction.

5.5.4 **Selections View**

To facilitate addressing task **T1**, we provide a mechanism to store selections of predictions. They are collected and presented in this view (see Figure 5.6D). Selections are depicted by indicating the occurrences of predictions for each class. To enable quick identification, a textual label is displayed together with the summary. Labels are defined by the user at the creation of a selection.

An indicator depicting the number of selected predictions is also shown. It reflects either an absolute or distinct count of predictions. The latter count is also used to compute a ratio over the total. The ratio is used to visually inform the user in case it goes above a threshold. The

threshold is determined by subtracting 100 and the model's accuracy percentage. This acts as a *warning* to keep the number of selected predictions low. The rationale for this threshold is that, assuming that the model's accuracy is similar to the one obtained for the evaluation data, then it should make around the same percentage of incorrect predictions in a real-life scenario.

5.5.5 **Input View**

Understanding why the DL model made a prediction (**T2**) is addressed with this view (Figure 5.6E). By visualizing an instance of input data (i.e., an epoch $E_{i,j,k}$), a resolution can be made to decide the correctness of a prediction (i.e., a classification $C_{i,j,k}$).

Our approach displays the values of signal f for a given epoch. Also, consecutive epochs can be displayed side-by-side at the same time to give a notion of context to the users. Furthermore, users can move forwards and backward to retrieve the next or the previous prediction's input respectively. This can be performed by pressing the right or left arrow on the keyboard. We visualize epochs through traditional line plots, with a fixed scale for the y-axis to facilitate the comparison of signals that have different amplitudes. Although this information is enough for the expert to infer the stage that an epoch represents, we also provide some clues on what the model was *seeing* at the moment of making a decision. To this end, our approach provides the corresponding values of a saliency map. By default, the saliency maps $M(C_{i,j,k}, l)$ and $M(C_{i,j,k}, l')$, where l and l' represent two convolutional layers from two convolutional branches of our model, are averaged. The layer parameter can be adjusted to visualize any convolutional layer of the model or a combination of them. Saliency maps are encoded as one-dimensional heat maps. This enables easy exploration of the salient regions of the input, without disturbing the visualization of the input data itself. This design addresses task **T2**.

Finally, the re-tagging task (**T3**) can be performed in this view. To quickly change the class of a prediction, glyphs are presented on the top of the view. Colors encode the class that a glyph represents. We also use position and text to encode some other information:

- The glyph at the left-most position indicates the current value associated with the prediction. In case it is the original prediction, it is marked with the label *P*, which stands for *Prediction*. If the class was changed by the user, the label *F* is used, which stands for *Fixed*. Moreover, the glyph also shows the probability that the model output by a text label.
- The other glyphs are slightly separated from the previous ones. This emphasizes that the probabilities they depict are normalized with respect to their values. This is helpful because the model we use tends to produce high-probability predictions. As a result, the probabilities for the rest of the stages are extremely low, making it hard to enable a comparison between them. By normalizing their values, easier comparison can be made by the user.

When a prediction is corrected, the visual encoding in all the other displays changes accordingly to indicate so. This is helpful to avoid the user from revisiting corrected predictions.

5.6 Use Case

Sleep is a natural process consisting of transitions between sleep stages that tend to follow rules. Alterations in the transitions can be an indicator of a sleep disorder. These alterations can be reflected in different manners: longer duration of a particular stage (e.g., stage *N1* should last up to 7 minutes), continuous changes between stages (e.g., from *N2* to *awake* and vice versa) or the absence of some stage (e.g., *REM*). These patterns might represent either the effect of a sleep disorder or problems in our predictive model to correctly classify sleep stages. The main goal of the exploration is to find out possible misclassifications and fix them.

We demonstrate our approach with a use case on a single dataset [81]. The subjects considered for our use case range from *SC4201E0* to *SC4822GC*. These subjects were not considered in the training phase of the DL model. Note that the source of the dataset also provides the ground truth, which we use in a posterior stage to calculate the percentage of *actual* misclassifications found in the exploratory use cases. We show how components in our approach together with domain knowledge from experts enable the identification of possible misclassifications as well as interesting patterns from the cohort.

5.6.1 Exploration Patient 1

The exploration (see Figure 5.6 for an overview) was conducted with the help of the somnologists. We picked a patient that seemed interesting because of the deviations in the distribution of sleep stages. The expert remarked that it was unusual to not have a single prediction of *REM* stage. This could be because the patient has some disorder, or because the model was wrong when making predictions.

REM stage usually happens after *N3*, or after a short period of *N2*. In Figure 5.2, we can see that *N1* and *REM* are somewhat similar in shape. Therefore, it may be that the model had difficulties distinguishing them. We started the exploration by narrowing the analysis to consider *awake*, *N1* and *N2*. We noticed some interesting patterns in the sleep cycles view. For instance, we saw that slightly two hours after the start of sleep, there was a noticeable drop in the probability prediction and there were some outliers from *N1* (see Figure 5.8 S2). We selected that region for further analysis. After moving forward in the sleep cycles view, we saw two other interesting patterns: quick alternations between *awake* and *N1*. This pattern was seen between four and five hours after the beginning of sleep (see Figure 5.8 S3). The pattern looked suspicious because of the quick changes between stages. We selected and saved them for analysis.

At the beginning of the sleep cycles view, there were two periods of *awake* followed by some predictions of *N1* and *N2* (see Figure 5.8 S1). We noticed that the probability of the model dropped in that region considerably. When we selected slightly more than the first hour of sleep, the dimensionality reduction plot showed some overlapping areas that looked suspicious. We selected and saved those areas. Nearly at the end of the sleep record, there was a period of *N3* predictions (see Figure 5.8 S4). According to the expert, this was suspicious because *N3* tends to shrink during the night. We selected all these predictions for further

analysis. Figure 5.9 shows some of the overlapping areas for selections *S1*, *S2*, *S3* and *S4*. The brushed predictions are potential misclassifications.

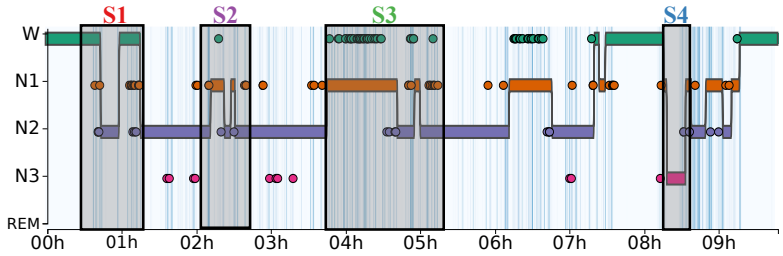


Figure 5.8: The four selections made in the exploratory use case.

At that point, we focused only on the dimensionality reduction plot to observe the whole picture. We observed sparse predictions of class *N1* and *N3* in an area principally covered by predictions of class *N2*. Therefore, we selected and saved predictions belonging to classes *N1* and *N3* in the overlapping area (see Figure 5.9 S5).

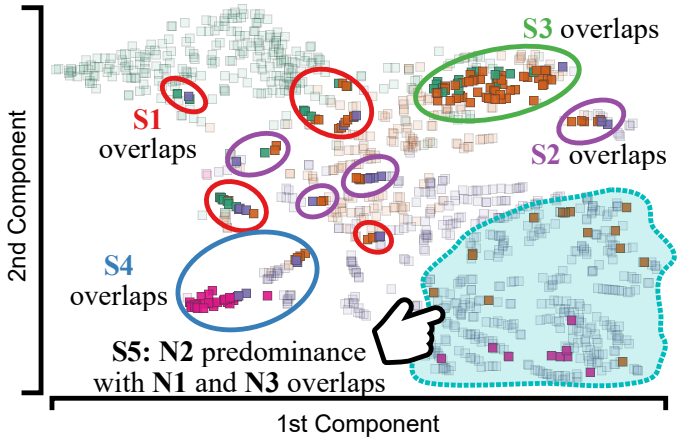


Figure 5.9: Selection of possible misclassifications. For instance, *S5* depicts an area of *N2* predominance, but *N1* and *N3* predictions are found.

After performing the selections, we ended up with 311 predictions. While reviewing the input data for each prediction, we annotated 217 as misclassified, which represented 69% of the whole selection. Posterior analysis using the ground truth showed that 249 predictions were misclassifications. Therefore, we found 87% of the misclassifications with our approach. It is important to remark that the information about the number of misclassifications was not available during the exploration of the data. During annotation of the block of *N3* predictions nearly at the end of sleep, we discovered that they were misclassified because the input signals seemed to contain an artifact (see Figure 5.10) following a very regular pattern. The expert indicated that this might be due to a problem with the location of the sensors that were interfering with the movements of the eyes.

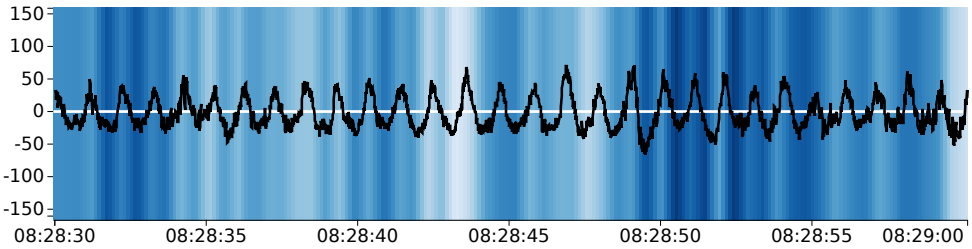


Figure 5.10: Artifact found during the exploratory process. It is too regular and free of noise to depict a biosignal.

The needed time to analyze the plots and to select pieces of data was about 5 minutes. We do not measure the time to analyze each epoch. However, if we consider the time proposed by the sleep scoring manual [119], which recommends investing up to 2 seconds per epoch, it would result in above 10 minutes. If we apply the same time per epoch for the whole output space, it would result in investing 39 minutes. Therefore, we would roughly save 24 minutes for this particular subject.

5.6.2 **Exploration Patient 2**

For the second exploration, we used the same patient and session as shown in the video demonstrating the usage of our approach. This session contained 775 predictions. We immediately observed that the sleep cycles seemed to be more uniform. It could also be seen in the locations of predictions in the dimensionality reduction view, which were better localized forming clusters-like structures.

In this case, we took advantage of the outliers from the sleep cycles view and the location of predictions in the dimensionality reduction plot. The sleep cycles view showed some interesting areas that contained outliers. They were seen after one hour and a half (a *N2* sequence with *N1* and *N3* outliers), slightly before two hours (a *REM* sequence with *N1* and *N2* outliers), around three hours (a *N2* sequence with *N3* and *REM* outliers) and so on. When selecting each of these areas individually, we observed the corresponding predictions in the dimensionality reduction view. We observed some overlapping areas in this plot. They could be an indicator of misclassifications, thus we selected and saved them.

After repeating this process iteratively, we created a global selection with 57 predictions. This represented 7% of the total output space. Out of those predictions, we found 35 misclassifications. Posterior analysis using the ground truth showed that there were 63 misclassifications. Therefore, we were able to find 55% of misclassifications. As for the previous patient case, this information was not available beforehand.

5.7 **Discussion and Limitations**

The use case depicted in Section 5.6 shows how our approach can be used by experts to utilize domain knowledge to guide exploration towards finding misclassifications.

Finding misclassifications when there is no ground truth is an unsolvable problem per se. Visually exploring and analyzing predictions can shed light and ensure a certain degree of correctness. The ability to select parts of data based on observations enables experts to incrementally cover most of the misclassified predictions. The tight interaction between components helps to verify earlier assumptions (e.g., quick transitions between stages might represent misclassifications).

As for other approaches, ours has limitations. For instance, the usage of our system does not guarantee the discovery of all the misclassifications. The interaction and exploration of predictions cannot always lead to finding all the faulty predictions, and in some cases, it might become difficult to understand the dimensionality reduction. Moreover, the dimensionality reduction might be ineffective if the model produced poor separations of the feature space. This limitation is not specific to our approach but inherent to the dimensionality reduction approach.

We performed an informal qualitative evaluation of our approach with both somnologists and DL experts. They found our approach useful and helpful to fill the gap in current settings in which a DL model is used. They expressed that being able to see the context of the predictions in different forms was very helpful. For instance, linking the time-location of a prediction with its location in the dimensionality reduction plot was useful to better analyze incorrect predictions. They also found views and visualizations of our approach appropriate for sleep experts. They stated that, after some explanation, the dimensionality reduction view was understandable and useful for spotting incorrect predictions. Also, the ability to create selections that matched hypotheses was an interesting way of addressing the problem. Finally, they also remarked that visualizing the input for a particular prediction in conjunction with the saliency map was useful to understand what the model was recognizing in signals. Having a smaller version of the sleep cycles view in the cohort view was proposed by one of the somnologists. This could help spot interesting patients from an overview of the transitions between sleep stages. However, it might be difficult to visualize due to the size of the components.

The somnologists also pointed at the lack of a mechanism to filter cases with some interesting properties. Filtering cases in which the input signal is mostly 0 volts would be desirable because it might reflect a misplacement or disconnection of the electrodes. The experts would also like to use the system to better understand the model. The interest of the experts arose when they observed the input of incorrect predictions. Even though this is not the goal of our system, we certainly believe this would greatly improve the system.

When the somnologists were asked whether they could use data corrected with our approach despite the fact it cannot be guaranteed to be fully correct, one stated the following: *"It really, really depends on where the errors lie. If there is some misclassification between N1 and N2, it probably will not affect the clinical interpretation too much. But if all errors converge*

to misclassifying REM as something else, it will affect it. As another example; if epochs are misclassified as wake sparsely through the hypnogram, it may result in a very fragmented looking hypnogram, which may be classified as abnormal; while if the same amount of (mistaken) wake epochs are grouped into 2 or 3 a little bit longer periods of consolidated wake, it may look normal to a doctor". We believe these issues are addressed by our approach since users can spot suspicious patterns, analyze them in terms of activations from the DL model and inspect corresponding input data. From the comments of the somnologist we can argue that the number of found misclassifications is not crucial as long those that could affect the diagnosis of sleep disorders are analyzed.

5.7.1 **Approach Generalization**

Deep learning models can be used to detect cancer tissue in video frames (e.g., colonoscopy [169]). Generally, the traditional approach works by examining a patient with a tubular camera that explores the colon while the doctor analyzes the video in real time. Deep learning can be incorporated in this process to aid doctors during the examination. Our approach can be generalized to be applied in this scenario. Sleep staging and cancer detection share the characteristic of having sequences of predictions of a recording session. The sleep staging model uses epochs, while the cancer detection model uses video frames. Although the nature of the data is different, they have the temporal aspect in common. Both tasks aim to classify input sequences in a small set of classes and both scenarios are divided by patients and sessions. To generalize our approach, it would need to handle video frames keeping the rest of the system intact.

5.7.2 **Scalability**

The bottleneck of our approach is the dimensionality reduction. Our implementation can compute tSNE over one thousand of records with thousands of dimensions in a few seconds. If the number of dimensions was higher, a pre-process step could be applied to randomly project dimensions to a lower space, feeding the result to the dimensionality reduction technique. This approach seems to be effective for tSNE as Donahue et al. stated [40]. In our approach, we handle over 1000 samples per patient on average. In case this number was too high to be handled, the analysis could be performed by examining chunks of thousands of samples per time. The major drawback of this approach would be to ensure that we have enough representatives of each class in each step. On the visual aspect, the main concern is the number of classes that our approach can present. In the case of sleep scoring, there are only five different classes. However, in other domains, this number might be substantially higher.

5.8 **Conclusions**

In this work, we have presented *V-Awake*: a visual analytics approach to find and correct faulty predictions in real-life scenarios. It is a novel visual analytics approach that combines different visual and interactive components to enable users to effectively find and correct predictions in a sleep staging context.

We have demonstrated the usability of our approach in a use case. It shows that our approach can be used to find suspicious patterns that can represent misclassifications. Besides, a generalization of our approach has also been proposed, stating that it can be transferred to other domains with minor modifications. The discussion with experts reflected a real interest from them in our approach as well as ways to improve it. Although our tool does not guarantee a perfect correction, it does enable experts to analyze interesting patterns to make sure that a proper diagnosis can be performed afterward.

As future work, we would like to investigate how to incorporate active learning such that users could reinforce the model with our approach. This idea requires more research to ensure that the learning process provides benefits instead of creating a bias in the model that deteriorates the predictive accuracy. We also plan to extend the approach such that users can explore the dimensionality reduction plot from a higher level, that is, focusing on the entire cohort population rather than a single patient. We expect this to provide some further insight into patients that have more faulty predictions. Our idea is that we could apply a dimensionality reduction method over the entire population to find groups of patients that share similar peculiarities in terms of activations of layers. With this, users would only have to analyze some representative subjects from a particular cluster and apply the learned facts to the rest (e.g., majority of faulty predictions in stage *REM*, similar sleep patterns, etc.).

6

Explainability for Sleep Staging

Most of the interpretability techniques for deep learning focus on visual representations of a very special type of input data: images. However, this is not the only domain in which neural networks are effective. Besides, several techniques currently exist to interpret the layers of a neural network. Most of these approaches are meant for convolutional neural networks (CNN) applied to images. As a result, there is a lack of contributions in the literature concerning interpretability for input domains different from images. In this work, we aim at shedding light on the challenge of explaining deep learning models' decisions for temporal data. Unlike images, time series are not that easy to understand. Even with the use of saliency maps to signal a specific region that the model considered important to make a prediction, it remains difficult to understand why the model considered that particular part of the input space more important. We explore some aspects of state-of-art approaches to see whether they are applicable to temporal inputs or not.

The contents of this chapter have previously appeared in **Garcia Caballero, H. S., Westenberg, M. A., and Gebre, B.** Explainability for one dimensional temporal inputs of deep learning models. *Demo at the 1st Workshop on Visualization for AI explainability (VISxAI)* (2018). Online publication [51].

6.1 Introduction

Most of the interpretability techniques for deep learning focus on visual representations of a very special type of input data: images. There are plenty of examples of deep learning architectures that try to perform a particular task on this input domain: object recognition [196], image captioning [24, 44, 185], image segmentation [118] or image classification [83], among others. However, this is not the only area in which neural networks are effective.

Much effort has been made to apply these models to other tasks like sleep scoring [163], speech recognition [8], music classification [25], to name a few. A new type of architecture has been proposed to deal with temporal data. The Convolutional, Long short-term memory, fully connected Deep Neural Networks (CLDNN) [141] can extract features from raw input signals using convolutional layers. Next, these features are fed to a Recurrent Neural Network (RNN), which takes into account the temporal relationships between the samples. Finally, a Deep Neural Network (DNN) produces the output of the model.

In terms of explainability, several techniques currently exist to interpret the layers of a neural network. Most of these approaches are meant for convolutional neural networks (CNN) applied to images. As a result, there is a lack of contributions in the literature concerning interpretability for input domains different from images.

Another important aspect when it comes to explainability is how to deal with architectures that combine multiple convolutional branches. The motivation to use more than one convolutional branch could be to extract different types of features from the input data. For instance, one part of the model could try to look at very small, local features, while the other might apply convolution on a bigger input range to extract global-like patterns. The resulting features that the model learned might not be easy to interpret in terms of input, since the model might be *seeing* different things in both branches leading to a misunderstanding of what the model learned.

Understanding the output of deep learning models in particular (and machine learning algorithms in general) might not be crucial in every field. However, in some contexts, this is an important fact that requires attention. For example, in a medical context, it is important to understand why a particular output was produced instead of another one. This may help to comprehend whether the model is performing in the right way or, on the contrary, it is producing the incorrect output.

In this work, we aim at shedding light on the challenge of explaining deep learning models' decisions for temporal data. Unlike images, time series are not that easy to understand. Even with the use of saliency maps to signal a specific region that the model considered important to make a prediction, it remains difficult to understand why the model considered that particular part of the input space more important. Showing such saliency maps can be misleading. Images are natural to humans because it is easy to recognize meaningful objects, and by highlighting some areas of the images, it is possible to associate ideas. On the other hand, signals are just values that fluctuate over time and might be meaningless. Thus, just signaling a region of these values may not be sufficient to form a mental idea that gives meaning to the pointed regions.

6.2 **Images and Time Series**

Images and signals are similar in terms of data representation. A signal of 100 points can be thought of as an image of 100 pixels wide and 1 pixel tall with a single channel. Although they are very similar in these terms, images and signals differ in their visual representations. While an image can be very quickly understood, a time series needs more visual aids to help the user to find interesting facts (e.g. trends, patterns, and so on). We can extrapolate from this fact into the explainability domain: while the explanation on an image is fairly easy to understand by any user, the same way of explaining may not be suitable for a time series.

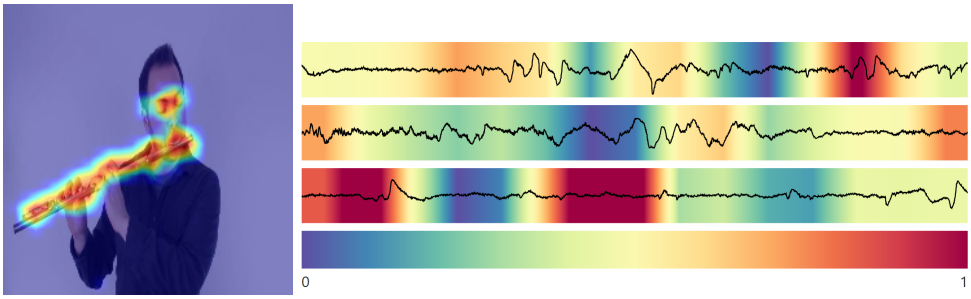


Figure 6.1: (a) shows an example of Saliency Map on image domain for VGG-19 model [149] where the model labeled the picture as "flute, traverse flute". The approach proposed by Fong and Vedaldi is applied to get the saliency map [45]. (b) shows a Saliency Map applied to three input signals for a model that scores sleep stages [163]. All of them represent the same class (*awake*)

Figures 6.1a and 6.1b represent two examples of saliency maps applied to both domains that we discuss in this work. Figure 6.1a depicts a person playing a traverse flute. The coloring on the image corresponds to the saliency map technique applied, which detects the traverse flute concept. We can see that the model looked at the music instrument itself, as well as the face of the player. This might mean that the model learned the *traverse flute* concept in pictures in which it was played by a musician. On the other hand, the examples shown in Figure 6.1b depict three different signals representing the same subjacent class (*awake*). It is quite difficult to understand the underlying reasons of the deep learning model to classify the three of them as *awake* by just looking at the saliency maps. The displayed saliency map corresponds to the so-called *l14_conv* layer of the model depicted in [163]. This model concatenates two convolutional branches. The layer we are using corresponds to the right branch, which is in charge of capturing frequency information. It can be seen that the first and the second examples have some similar patterns. However, the model focuses on some part that does not look particularly relevant at first sight. Moreover, the third example seems to give the highest importance to a different wave shape.

6.3 **Case Study**

6.3.1 **Model**

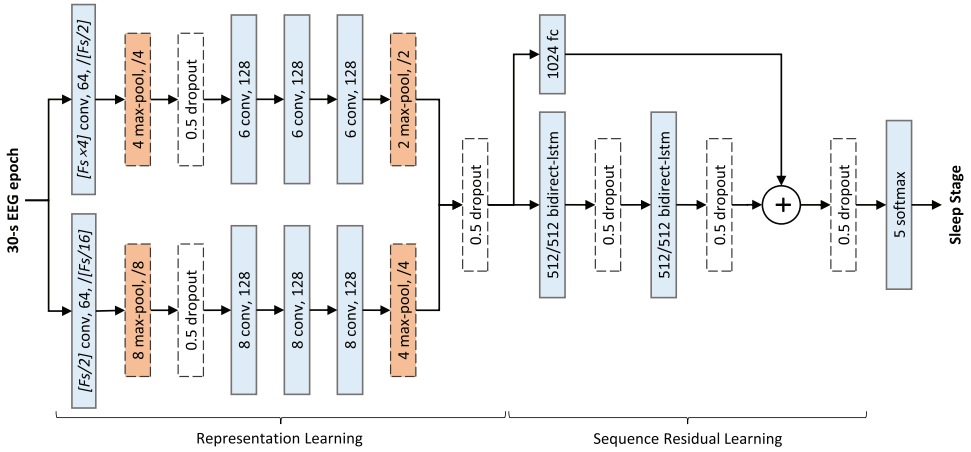


Figure 6.2: Architecture of the model used in this work (taken from [163]). Trainable layers are depicted in blue.

The model used [163] to conduct this case study is shown in Figure 6.2. The model is used to score sleep stages from a raw input signal. It has a CLDNN architecture with two convolutional branches that are concatenated and fed to a bidirectional long short-term memory layer. A residual learning approach is also used in this model to maintain the features extracted by the convolutional layers of the model. Finally, these features are added up element-wise with the ones generated by the two bidirectional LSTM layers to be fed to a softmax layer that produces the final output, which can be either awake, S1, S2, SWS and REM. This class represents the sleep stage in which a patient is at a certain point.

The data used to train the model [56] represents the sleep recordings of 20 patients for two nights. Those recordings are taken from sensors placed on the head of the patients during sleep to record the so-called electroencephalograph. The model that we use was trained with the *Fpz-Cz* derivation. An overview of possible placements for the electrodes in the electroencephalography can be seen in Figure 6.3. The data per patient and night forms the input of the model, which is divided into 30 seconds long epochs. The sampling frequency is 100*Hz*, resulting in 3000 points per epoch. The data was obtained between 1987 and 1991 in a study of age effects on sleep. Each recording night has 1075 samples on average. That is, there are 1075 predictions per patient per night on average. Input examples are shown in Figure 6.4 to provide a reference point on how the input space looks like. All examples belong to subject SC4022E0 (second night).

Sleep experts usually look at multiple derivations to score a given epoch. However, this model was trained with a single derivation and still scores quite accurately compared to experts. That is why it seems interesting to investigate the decisions that such a model is taking

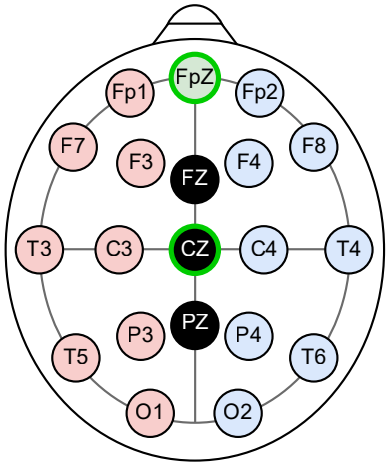


Figure 6.3: Illustration of possible placements of electrodes to record electroencephalography. In this work we are using the placements indicated with the green stroke.

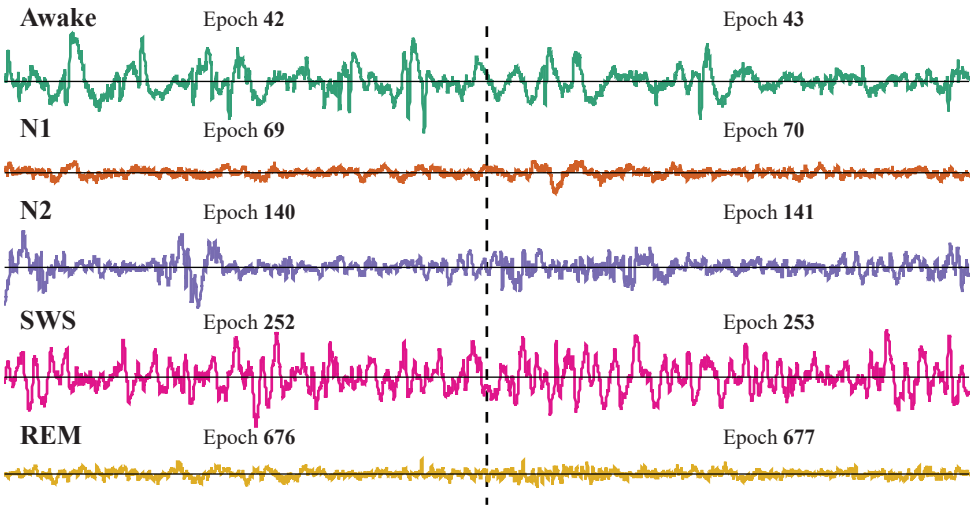


Figure 6.4: Fpz-Cz derivation for several epochs from the second night of subject SC4022E0. Each row depicts a distinct sleep stage according to the ground truth score.

when processing epochs. More information on sleep scoring can be found in the American Academy of Sleep Medicine manual for the scoring of sleep [119].

6.3.2 **Dataset**

The data used in this section is a sample, which represents a patient, of the previously described dataset. The model has not seen that sample in the training stage. Moreover, the

epochs that form the sample have been selected randomly to ensure they are as heterogeneous as possible. We have selected three epochs per class, as well as others that the model classified incorrectly.

6.3.3 **Model Explanation**

In order to explain what the model is doing to classify instances, we are going to use Grad-CAM as a saliency map technique. Grad-CAM [144] is a technique to produce *visual explanations* of CNNs. The technique is a generalization of the work of Zhou et al. [200] in which no modifications on the model are needed to obtain the saliency map. This approach takes the gradient information into account to determine how the neurons of a CNN layer contributed to a specific-class prediction. This information is usually then rendered as a heat map on top of the input to indicate the contribution of that subset of the input to the final output.

The visual interface designed for this use case provides a widget with the possible classes that our model outputs. By selecting a class and a layer of the model we can visualize the Grad-CAM coefficients as a gradient. In principle, these coefficients depict the parts that the model *considered* important to output the prediction. Notice that the output of the Grad-CAM is layer dependant, thus the output of Grad-CAM will have the same shape of the output shape of the layer. We have applied an interpolation to re-scale this size to match the input size (3000 points). This widget also enables the selection of misclassifications to be visualized.

Generally, in the models with just one convolutional branch, the most interesting layer to look at is the last convolutional layer. It is believed that the deeper layers of the model tend to capture higher-level constructs. Thus, in our case, we have two main options (see Figure 6.2).

The bottom convolutional branch of the model has a smaller filter size (FS/2). This can be seen in the result of the Grad-CAM for the left layer. Apparently, it pays attention to more local and prompt features. On the other hand, the top convolutional branch of the model has a bigger filter size (FSx4). Because of this, the visualization shows longer parts of the input that the model looks at. This layer seems to have somewhat recognizable patterns. However, it is not always the case.

An interesting effect can be seen when inspecting the *REM* or *SWS* stage for the last convolutional layer of the larger filter branch (see Figure 6.5). It can be seen that the first sample of REM and the first and second samples of SWS have only zeros as Grad-CAM values. This could mean:

- Either the input only produced negative activations, which are not considered in the Grad-CAM (notice that it uses a ReLU approach to remove the negative activations); or
- the input only produced zero activations, meaning that this layer has no filters capable to recognize any particular pattern.

Moreover, the second and third REM samples show a somewhat similar pattern, and just a small part of the input is considered to be important. Interestingly, it is exactly the opposite

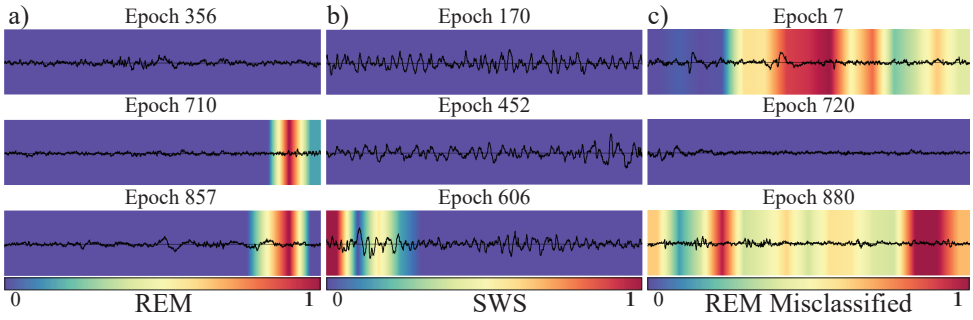


Figure 6.5: Grad-CAM coefficients for different sleep stages and epochs. (a) depicts examples for REM sleep stage, (b) for SWS and (c) for samples that have been misclassified as REM.

for those samples that have been incorrectly classified as REM. This might highlight that the model needs no features from this layer to classify correctly samples for those two stages.

If we continue with the same class (*REM*), and now look at the last layer of the smaller filter size convolutional branch (see Figure 6.6), we can see a more uniform distribution of the coefficients. This could mean that in order to classify a sample as REM, the model mainly uses the left convolution. However, it still remains unclear why some parts are considered of more importance than others.

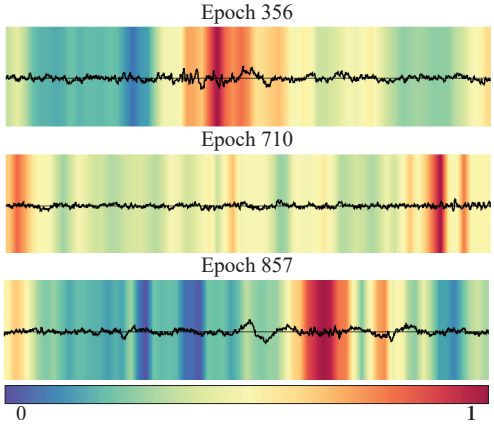


Figure 6.6: Grad-CAM coefficients for epochs correctly classified as REM. A uniform distribution of coefficients can be observed.

6.3.4 Does the model predict the same by occluding the input?

We have seen that by applying Grad-CAM some parts of the input get strongly highlighted compared to others. It seems logical to wonder what the model would do if we occluded some parts of the input. In this section, we will hide portions of the input to see whether the

model still produces the same output or not.

In order to make this case study more reliable, we will occlude just one epoch per time to ensure that the LSTM layers do not affect the output.

To hide part of the input space we consider the following aspects:

- **Grad-CAM values:** These values will be used to determine what parts of the input space must be hidden.
- **Occlusion direction:** We think of two manners to hide values: **1)** values that are less than the set threshold; and **2)** values that are greater than the threshold. The first will start hiding from the lower coefficients (i.e. zero). The latter will start from the highest ones. This is analogous to beginning with non-important values or the most contributing ones.
- **Occlusion operator:** When hiding parts of the input, we usually cannot tell the model that it should not consider them. That is why we need to define an operator that generates new values to replace the original input. In this work, we have used three, but others might be used: zero, random, or sinusoidal values. Except zero values, the other two operators map values in the range -158 and +158. The sinusoidal operator has been chosen because it well represents a basic, standard wave.

If we start tweaking these parameters by first interacting with the "less than" operator, we immediately notice that the model produces different outputs with some minor modifications. Figure 6.7a shows a selection of epochs that were predicted as REM. The background depicts the coefficients of Grad-CAM for the last convolutional layer of the bigger filter branch. If we try to occlude parts of the input with low Grad-CAM coefficients, we are *occluding* the values that are, in principle, not important for making the prediction (see Figure 6.7b). However, we see that the model is predicting something different (awake) for epoch 720. This might be because we are occluding values that are considered important by the right layer. However, by switching to the last layer of the smaller filter branch (see Figure 6.8a) and using the same configuration, epoch 7 gets classified as awake and epoch 880 as N2 although we are just occluding their least important portions of the input space (see Figure 6.8b).

We have noticed that the occlusion technique cannot be applied in the same way as for images. When dealing with images, the occlusion happens by setting the pixels of a sub-region of the image to a constant value. In the case of time series, we can decide to set the value of a range of the time series to 0, random values or whatever other function that replaces the original values of the series. The main drawback appears because the model seems to be able to recognize those new values as another class. Therefore, depending on the occlusion operator the model will output different predictions even when we replace the values that are not important for a given layer of the model.

Another interesting fact is that some operators seem to have a stronger effect than others. For instance, the random operator seems to force an awake prediction with minor changes on the input. Zero and sinusoidal operators seem to have a similar strength. However, the output tends to go towards different classes.

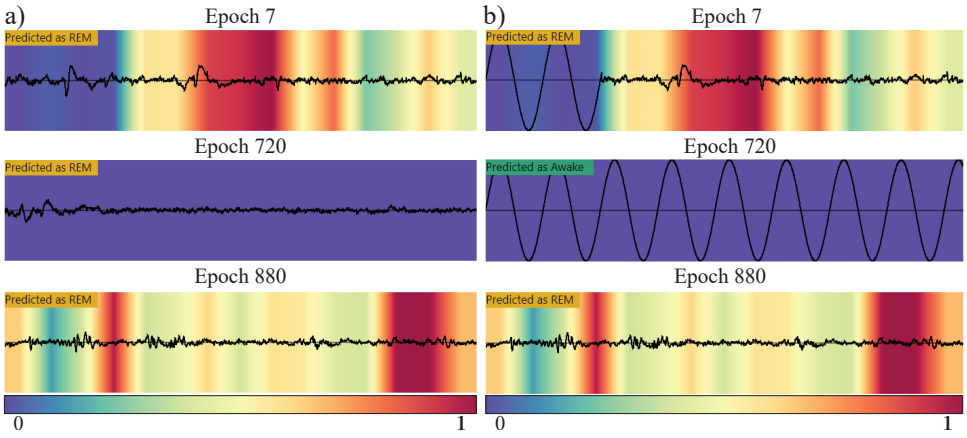


Figure 6.7: Examples of occlusions applied to three epochs classified as REM. (a) depicts three epochs with their corresponding Grad-CAM coefficients for the last layer of the bigger filter branch, whereas (b) shows a sinusoidal occlusion applied to the lowest Grad-CAM coefficients. Epoch 720 results in a different prediction.

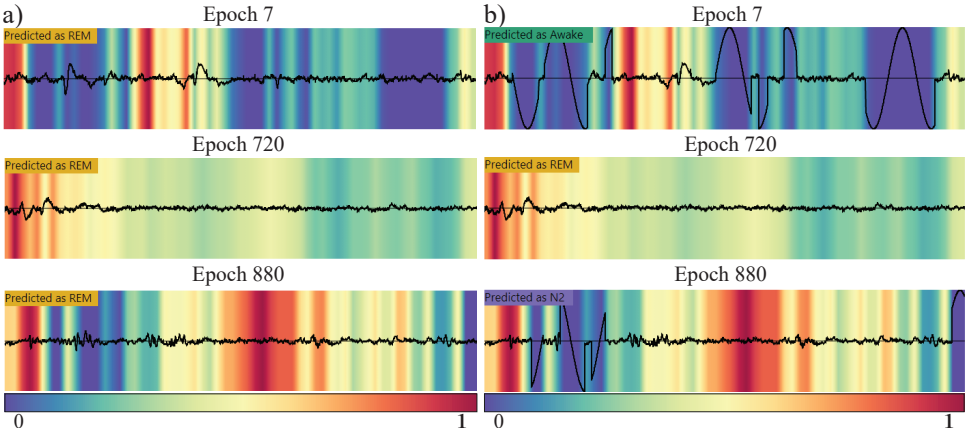


Figure 6.8: Same examples as in Figure 6.7 but (a) Grad-CAM is now applied to the last layer of the smaller filter branch. (b) applies the same occlusion technique resulting in different predictions for epochs 7 and 880.

6.3.5 **Learned Filters**

In this section, we will explore the filters that the model learned in the training phase. In order to do so, we look at the activation maps produced by the layers of the model. Intuitively, the same filters should be active for samples of the same class.

The widget below shows the filters of the selected layer using squares. The color of the square depicts how active that filter was for the given input, darker being more active. The function that determines how active the filter was can be tweaked with one of two options: average or maximum. The first takes the average value of the activations, and the second takes the maximum value. Filters can be hovered to get a plot of the activations. When a filter is

hovered, the same filter is also plotted for the other samples. By clicking on the filters, they will remain highlighted, allowing to explore the activations and see what part of the input was responsible to generate such activation. Notice that only the two first layers of the model can be explored in the activations plot on the left of the samples. This decision has been made because it makes sense to map those activations to the input space, but it does not for deeper activation maps that are computed over the activation maps of previous layers.

In general terms, we have noticed that the filters of the left branch of the model tend to produce similar activations for the same class. On the contrary, filters on the right branch appear to be somewhat more sparse.

Let us take a closer look at the stage Awake and the first layer of the smaller filter branch. In Figure 6.9a we can see three different epochs and the filter activations for the aforementioned layer, where black depicts a *strong* activation and white depicts no activation. We observe that similar groups of neurons tend to be active. If we explore one of the black filters we can easily discover that their most intense activity occurs in parts of the inputs that are similar for all the samples (Figure 6.9b). On the contrary, if we look at the white filters, no interesting pattern is visible (Figure 6.9c).

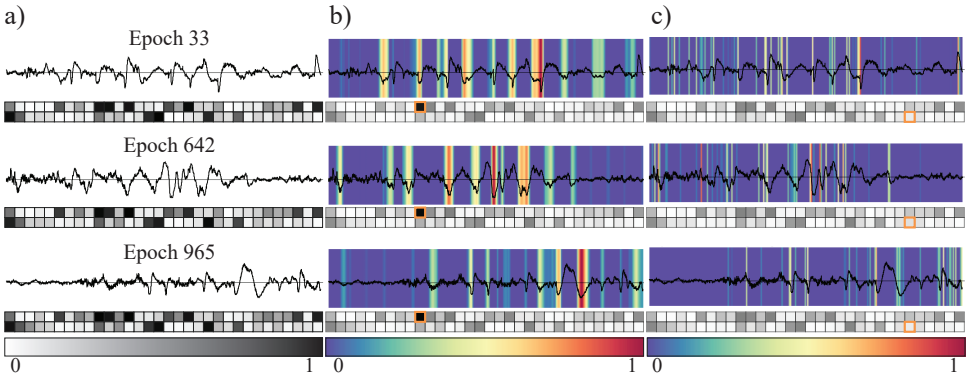


Figure 6.9: (a) Learned filters of the first layer of the smaller filter branch. (b) shows the activation in the input space for a an active filter, whereas (c) depicts a nearly non-active filter.

In some cases, some filters are rarely active for all the samples that we consider in this case study. This might mean that they have not learned any feature in the training phase that can be applied to our samples.

If we move now to stage S1 and we combine with the first layer of the bigger filter branch (see Figure 6.10a, big filters), it becomes apparent that it is slightly harder to see possible patterns that the model is recognizing because this type of wave has a very small variation. Also, we notice that the activation of the filters is not as homogeneous as in other cases. On the other hand, the first layer of the smaller filter branch (see Figure 6.10a, small filters) has many more similar activations for the three samples of this sleep stage. Also, if we explore the activations for the most activated filters we can see that they are active in parts of the samples that have a similar pattern. For instance, we can see that one filter (see Figure 6.10b) is detecting parts of the signal that are local *minima*, while another filter (see Figure 6.10c)

seems to look at local *maxima*.

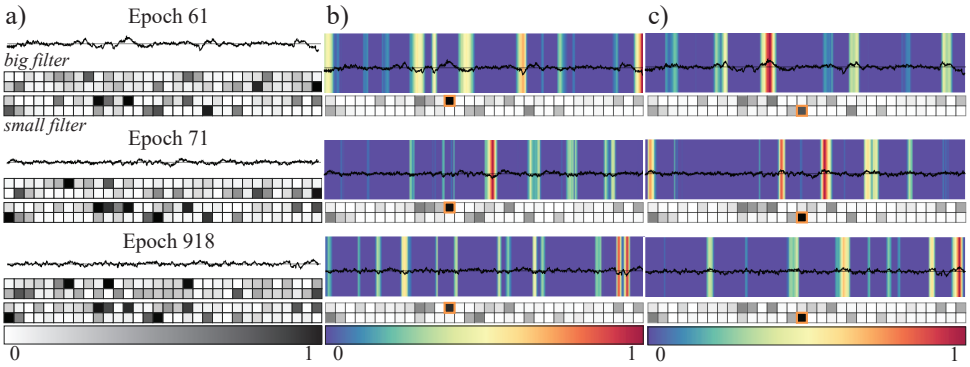


Figure 6.10: (a) Comparison of learned filters for the first layer of both big and small filter branches for three different epochs, (b) a filter that seems to recognize local *minima*, and (c) a filter that recognizes local *maxima*.

Another interesting fact can be seen by alternating with misclassification. In this case, we can see that different filters get activated for the right classifications and for the wrong ones. This is somewhat reasonable since it indicates that the model has recognized a class by using different filters. This might be used in other contexts to help to identify wrong predictions of a model.

6.4 **Discussion**

During this work, we have seen that saliency map techniques applied to time series can indicate what the model was looking at to some extent. However, we think this is still insufficient to understand models that deal with temporal data.

An important fact that we have not considered in this work is how the recurrent part of the model influences the predictions. We believe it is important to develop explanations that combine both parts since they are related. However, it is still not clear how to address this challenge. One may think that it is important to understand what features are being extracted by the convolutional layers of the model. Nevertheless, one could also think in terms of relations between sequences of inputs, or sequences of features from the inputs, as recurrent layers do. The combination of both views might play a crucial role in the understanding of the way the model is making predictions and not solely considering convolutional or recurrent layers apart from each other.

While it is still unclear how to deal with our challenge, we believe there may be some basic approaches that could, at least, help final users understand what deep learning models are doing inside when they deal with temporal inputs. For instance, providing users with basic information such as averages of certain regions of an epoch, main trends or correlation coefficients, to name a few. The combination of these indicators may lead to a better understanding since the traditional techniques just provide us with some internal information of the model reflected on the input.

Final users of the deep learning models represent another important factor that must be considered when developing new explanation techniques. In some cases, they are not familiar with the model itself and they lack knowledge in the area of machine learning. However, they can benefit from using these models to boost their daily processes, perform tasks more accurately or just to get extra support in their daily challenges. For example, the medical domain can (extremely) benefit from using deep learning models, although some effort is needed to *unbox* these types of models and explain them in a simple, accurate way.

6.5 **Conclusions**

In this work, we have introduced the problem of making sense when dealing with temporal inputs in deep learning approaches. As opposed to images, temporal inputs are not as easy to understand as images. Thus, understanding what a deep learning model is ”looking at” when making decisions is also hard.

The aim of this work is at triggering curiosity in the community of deep learning and visualization to explore alternatives that effectively deal with temporal inputs (e.g. brain signals, voice, market stock fluctuation, etc.), different architectures (e.g. multiple convolutional branches) and the combination of convolutional and recurrent layers. All these ingredients shape a gray area that, to the best of our knowledge, has not been deeply studied yet. Although several techniques have been developed recently to interpret these models, they are mainly focused on images rather than other types of inputs. On the other hand, the usage of convolutional layers in conjunction with recurrent ones is becoming more and more popular to deal with temporal data. In this scenario, the convolutional branch acts as a feature extractor that are then fed to the recurrent part of the model to make the decision. Therefore, effectively understanding what the model is doing is a crucial step to get reliable models that can be used in somewhat more critical tasks (e.g. diagnosis).

7

Performance Assessment of Sleep Staging Models

Machine learning is becoming increasingly popular in the medical domain. In the near future, clinicians expect predictive models to support daily tasks such as diagnosis and prognostic analysis. For this reason, it is utterly important to evaluate and compare the performance of such models so that clinicians can safely rely on them. In this paper, we focus on sleep staging wherein machine learning models can be used to automate or support sleep scoring. Evaluation of these models is complex because sleep is a natural process, which varies among patients. For adoption in clinical routine, it is important to understand how the models perform for different groups of patients. Moreover, models can be trained to recognize different characteristics in the data, and model developers need to understand why and how performance of the different models varies. To address these challenges, we present a visual analytics approach to evaluate the performance of predictive models on sleep staging and to help experts better understand these models with respect to patient data (e.g., conditions, medication, etc.). We illustrate the effectiveness of our approach by comparing multiple models trained on real-world sleep staging data with experts.

The contents of this chapter have previously appeared in **Garcia Caballero, H. S., Corvo, A., van Meulen, F., Fonseca, P., Overeem, S., van Wijk, J. J., and Westenberg, M. A.** Persleep: A visual analytics approach for performance assessment of sleep staging models. In *Eurographics Workshop on Visual Computing for Biology and Medicine, VCBM 2021* (2021), The Eurographics Association [49].

7.1 Introduction

Machine Learning (ML) has increased in popularity in the medical domain [117] due to its success in tasks such as segmentation, classification and anomaly detection. One example is sleep medicine, where models have been proposed to score sleep stages and support sleep diagnosis [127, 158, 163]. These advancements bring opportunities to automate such time-consuming [72], tedious and subjective tasks typically conducted by specialists. Assuring a good performance of such models is crucial for somnologists to safely rely on them.

Generally, the evaluation of ML models for sleep staging is complex for three reasons. First, sleep is a natural process that runs and evolves over *time*. When predictions are produced by a model, errors can occur at different periods of the sleep. The location of these errors is crucial because it can bias the diagnosis of sleep diseases (e.g., non-REM parasomnias usually occur in the first third of the night). Second, common statistics (e.g., accuracy, F-measure, etc.) only provide a coarse-grained perspective of the performance [198]. A closer look at predictions and patients is necessary to better evaluate ML models. Finally, wrong predictions can suggest that fragmented sleep occurs. This can potentially be misinterpreted as a sleep disorder. Inherently, all these problems can be different across *groups of patients*. Sleep varies among patients due to physiological reasons (e.g., age, medication, etc.). Therefore, models can be faulty in generalizing among different groups of patients. A more personalized approach would be beneficial to better understand the behavior of ML models among different groups.

In recent years, the availability of different forms of data has enabled the construction of ML models that exploit different characteristics of the data. In particular, we observe a trend towards usage of so-called surrogate devices such as smart watches, phones, etc. as the source of data for ML models [47, 95]. In the case of sleep staging, surrogate devices can be used to track the sleep of patients for longer periods and in less-intrusive manners than polysomnography. Usually, models consuming surrogate data output a smaller set of sleep stages than those trained on polysomnography due to less detailed data (e.g., high detail electroencephalogram (EEG) vs. low detail actigraphy and heart rate). In general, models trained on different data can fail in recognizing situations such as sleep fragmentation, arousals, etc. Analyzing and comparing models for sleep staging with heterogeneous sources of data can provide valuable insights to experts.

To the best of our knowledge, no approaches have been presented yet to conduct performance analysis in this sort of scenario. To this end, we present PerSleep, a visual analytics approach that aids ML experts in sleep staging to assess the performance of the models they employ. Our main contribution is the first visual analytics approach to evaluate and compare performance of two models in sleep staging. Multiple hypnograms can be visualized simultaneously to make quick comparisons of the same target hypnogram for different models. In comparison with state-of-the-art approaches in performance analysis, ours does not solely focus on the input data of the model but also the patient data. The novelty of our work lies in the application of visual analytics in sleep staging rather than the design of new visual idioms. Furthermore, we hope that our work provides a useful example for the assessment of complex models for judging time series data for varying populations, like neurological

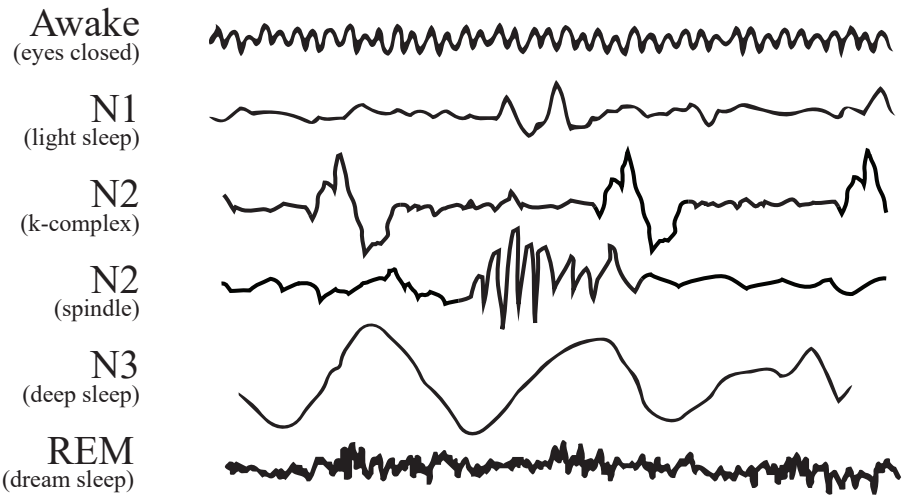


Figure 7.1: Examples of EEG waves and their corresponding sleep stage. Time and amplitude scales are different in each example.

brain disorders [6] such as epilepsy and autism, or physiological disorders like heart failure detection [87]. We present a use case on real-world data to demonstrate our approach. The use case was conducted with three experts. Results, limitations and generalization are discussed. Finally, we provide directions for future work for performance evaluation in sleep staging.

7.2 **Medical Background**

When a patient is believed to be suffering from a sleep disorder, a *polysomnography* (PSG) is often done, an electrophysiological recording of sleep and sleep-related events overnight. In a PSG, brain activity (measured by EEG), muscle activity (measured by EMG) and eye movements are recorded to assess sleep structure. In addition, other aspects are recorded such as body movements and breathing patterns. Afterwards, the recording is evaluated and annotated on an *epoch-by-epoch* basis. *Epochs* represent time-fixed periods (typically 30 seconds duration), which can then be analyzed by technicians in order to assign a sleep stage. The process of assigning sleep stages to epochs is called *sleep staging*, which was invented in the 1960's. During sleep staging, sleep is scored epoch-by-epoch as one of the five disjoint categories according to the American Academy of Sleep Medicine (AASM) [119]: *Wakefulness*, *N1*, *N2*, *N3* and *REM*. Sleep stages are characterized by specific physiological properties, which are based on consensus criteria. Figure 7.1 depicts some characteristics that can be observed in the EEG signals of a PSG. *Wakefulness* with the eyes closed is usually characterized by *alpha* waves (8-13*Hz*) in the EEG produced by the occipital lobe of the brain, while sleep stage *N1* often presents *theta* waves (4-7*Hz*). Other stages are characterized by interactions of multiple physiological stimuli that result in the presence of EEG phenomena

like k-complexes, spindles, or sawtooth-like waves. The sequence of annotated sleep stages during sleep is visually represented by a *hypnogram*, which is analyzed by a somnologist to understand the sleep pattern of a patient.

Generally, somnologists look for patterns in the hypnogram in terms of overall presence of and transitions between sleep stages. These patterns have clinical meaning, i.e., they can be indicators of sleep disorders. For instance, *fragmented hypnograms* present many transitions between sleep stages occurring in short time intervals, resulting in a fragmented sleep pattern. Such pattern can be indicative of a sleep disorder such as insomnia and narcolepsy.

Progress made in ML in this area has brought the opportunity for hospitals to switch to automated methods, which can be used to score PSGs. To this end, models need to be robust and reliable to assure the validity of the outcome they output. To support their assessment, better understanding of how models perform on clinical data is essential, and with our work we aim to contribute to that.

7.3 **Problem Definition**

It is difficult to develop ML models with a high accuracy and reliability in sleep staging. Furthermore, assessing the performance of a model is non-trivial. The result of an automated process is a hypnogram, rather than just statistics on individual epochs, and the overall quality of a hypnogram is hard to assess. Also, the performance of a model can depend on characteristics of the patients, such as age and gender. In general, models in sleep staging do not consider the demographics of the patients and focus on the physiological signals such that the model can generalize from these. This is due to two reasons. First, if demographics such as age and sleep disorders were to be considered by the model, it would require an immense amount of representative data of all combinations of age groups, and each sleep disorder, requiring thousands of participants. Obtaining this large amount of data is challenging and time consuming as it often involves patients being recorded, for at least one entire night, with many sensors in a sleep center. Second, it may be that the sampling done when selecting the patients introduced bias due to specific features (i.e., artifacts) of such selected group being hooked on by the model. These may not be true for other patients belonging to that category and not included in the sampling. The goal of this project is therefore to develop a visualization to enable experts to evaluate and understand the performance of ML methods for producing hypnograms, also in relation to the properties of patients.

The data used in performance assessment of sleep staging models is multivariate as it combines patient's data of different nature (e.g., demographics, clinical information and physiological records) and the model's data (e.g., predictions and probabilities). The dataset depicting patient data is a *table* with an undetermined number of attributes. Typically, age (quantitative), gender (categorical) and other comorbidities (categorical) are part of the patient's data. In general, both categorical and ordered attributes can be part of the patient's data. Moreover, each physiological record of the patient is a *field*, where each cell depicts the physiological measurements for a given point in time. In sleep staging, the sampling frequency is uniform. In most situations, this field dataset is fed to the ML model to predict

a *list* of sleep stages. Each sleep stage is a categorical value representing one of the possible classes defined by the AASM. Similarly, a list of probabilities is also produced by the model, where each probability is a quantitative value. Aggregations are often used to summarize performance data. For example, a confusion matrix is a *table* where both items and attributes depict sleep stages. Each cell of this table contains a quantitative value.

The system should support various levels of detail for the evaluation of the performance, where each level leads to its own questions, and enable smooth transitions between these:

L1 Based on individual epochs: What is the *probability* of the model for a given prediction? What was the *input data* for a given epoch?

L2 Based on individual hypnograms: What are the *main differences* between two models? How did the *confidence* of the model fluctuate over the entire night?;

L3 Based on aggregate results across large sets of patients: What are the *scores* for aggregate statistics? Are there *correlations* between data attributes?;

Furthermore, the expert must be able to split the set of patients into cohorts, specific subgroups, based on their properties, and compare the performance for cohorts and focus on specific cohorts, to answer questions such as *how does one subgroup compare to another?* and *are there groups of patients that have similar performance indicators?*. Also, rather than focusing on just a single model, the expert should be enabled to compare multiple models using different data and/or ML models where the *test* model depicts the one to be evaluated (e.g., neural network) and the *reference* model acts as ground truth (e.g., manual scoring).

From the previous levels of detail and questions, we derive the following tasks:

T1 Explore the distribution of patients in terms of attributes. This provides an overview of what sort of distribution an attribute follows for the entire group of patients. Visualizing such distributions can help to detect odd behaviors in our model. [**L3**]

T2 Find correlations between data attributes. Correlations are important to gain insights into the behavior of the model. For example, it may be the case that our model performs worse for patients that are old and take a specific medication. Hence, experts should be enabled to perform selections on attributes to generate and validate hypotheses. [**L3**]

T3 Analyze the performance of a model. Summarized statistics such as accuracy or kappa, only give a glimpse of the whole picture. Instead, an exploratory process is needed to gain insights into several factors that usually are intertwined. For example, accuracy value can be low and yet the clinical interpretation of both test and reference hypnograms be the same. [**L3**, **L2**]

T4 Compare hypnograms. Visualizing the hypnograms for both test and reference models is crucial to understand whether the performance of the model is good enough for medical purposes. When inspecting the epochs of a patient, the approach must enable experts to select portions for closer inspection. Input data should be provided to contextualize an epoch. [**L2**, **L1**]

Tasks **T1** and **T2** shall also be performed for groups of patients.

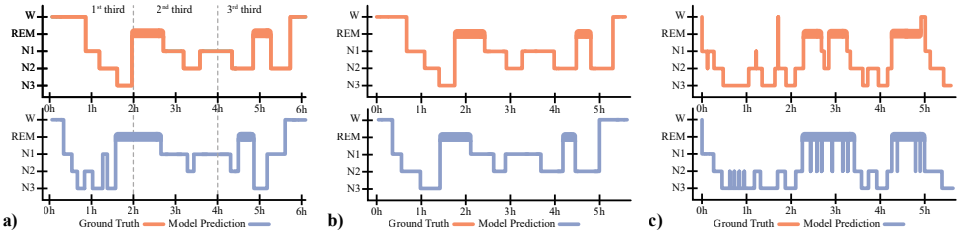


Figure 7.2: Illustration of three example problems in performance analysis for sleep staging: error grouping (a), global performance (b), and fragmentation (c). In (a), most of the misclassifications occur in the first and last third of the night, whereas the second third is the most accurate. (b) shows shifted predictions where the model predicts nearly the same global pattern but slightly earlier in time, resulting in low accuracy. Last, in (c) we see many transitions (*fragmented hypnogram*) between REM to N2 that are not present in the ground truth.

7.4 **Related Work**

In this section, we provide a review on previous work on performance analysis, time series and sleep analysis.

Performance Analysis. In performance analysis, predictions are the core element to be investigated. Generally, they are generated in combination with a set of probabilities that indicate how likely the prediction is to be of a certain class. A common approach in performance analysis is to explore the entire set of probabilities to find possible outliers. Most approaches visualize probabilities grouped by predicted class. ModelTracker [7] does not stratify predictions in classes because it just considers binary classification. Squares [136] is an extension from ModelTracker to support multiclass analysis. It makes use of histogram-like visualization to show probability distributions. They explicitly divide these into two groups: labeled and predicted class. The authors use color encoding to depict situations where both labeled and predicted class agree or disagree. Our work follows a similar approach as Squares, but using a somewhat simplified visual encoding to present probabilities. Moreover, we complement it with a confusion matrix that is used to explore specific cases in more detail by means of interaction.

Boxer [55] is a system that assists experts in developing and assessing classifiers. They address multiclass classification by means of interactive views that are formed by standard visualizations. It allows experts to layout views in *boxes* such that it gives different perspectives of the data, resulting in a flexible analysis of the performance of the classifiers. While Boxer considers multiple classifiers at the same time, our approach focuses on two models to maximize contrast and highlight differences. Furthermore, Boxer does not handle time-series data. In addition, our approach is model-agnostic within the sleep staging domain.

Time Series. A common visualization approach consists of displaying the input data of a model together with the predictions. It enables the exploration of the input features of a model. In the sleep staging domain, the input data is temporal. This data has received little attention as most works in ML literature target either multidimensional [198], text [160] or image data [126]. All these approaches provide interaction to select subsets of predictions to explore the whole (or partial) input space.

Some work has been done in understanding ML models where time is a component. RetainVis [86] and DPVis [85] utilize neural networks and continuous-time hidden Markov models respectively, whereas our work is model agnostic. They stratify patients into different groups defined beforehand, wherein our approach any data attribute can be used for this purpose. Finally, RetainVis and DPVis take feature contribution as key in their designs. Our work, however, does not take feature contribution into account and just focuses on performance indicators and patient data.

Sleep Analysis. The work of Combrisson et al. [28] presents a visualization tool to help technicians to manually score hypnograms. Automated techniques are used in their work to detect characteristic features in EEG signals. Their approach emphasizes the detected features in the polysomnography such that technicians can make better informed decisions when scoring the hypnogram. Although we do not aim to manually score hypnograms nor detect features in the EEG, we do make use of hypnograms in our approach. Hypnograms are the *de facto* standard of visualizing the sequence of sleep stages, and are also used in several other approaches [46, 127]. Our work focuses on assessing the performance of models, whereas the aforementioned work addresses the design of ML models from a pure ML perspective focusing on the inner workings of models.

In earlier work, we introduced V-Awake [52], a visual analytics system to help sleep technicians finding potential misclassifications from deep learning models in sleep staging. V-Awake was received positively by the domain experts, but rather than fixing errors afterwards, they also aimed to develop better models, and they missed systems to assess the quality of these. This led to the work presented here. PerSleep would be used when designing a model. Afterwards, V-Awake would be used in a real-world scenario to find possible misclassifications.

In summary, the major contribution of our work is the first system that focuses on the evaluation of the performance of models for sleep staging. Sleep staging is very special and important, and we must carefully design something ad hoc for that. It differs from other domains due to the incorporation of a time domain (epochs) and the need for flexibility to define subgroups. Currently, systems do not consider both characteristics simultaneously and generally focus on predictions independently from each other. Also, sleep staging is a great example of a ubiquitous pattern, not only for health applications, but also many other domains like epilepsy and autism [6], and heart failure detection [87].

7.5 PerSleep

The tasks defined in Section 7.3 steered the design of PerSleep. In this section, we introduce its main components (see Figure 7.3).

7.5.1 Model Selection

The model selection component (Figure 7.3A) enables users to import new models (containing scoring data) to PerSleep by clicking the plus button. These models can be easily selected

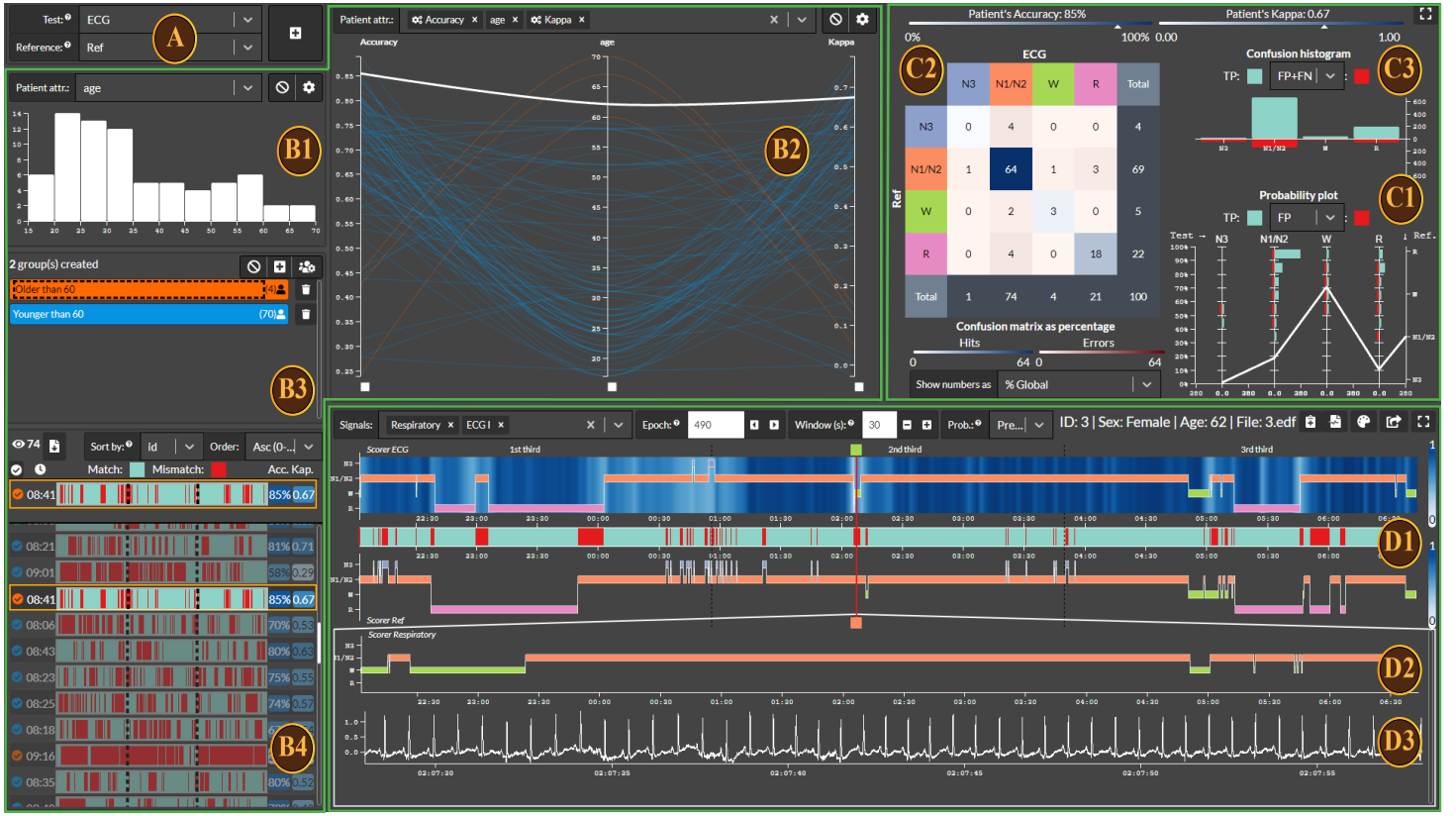


Figure 7.3: The views B1-B4 are patient-related, where B1 and B2 depict views where patients can be filtered based on the selections performed. B4 displays the patients that match the current selection. Groups of patients can be created, modified and selected in B3. C1-C3 shows performance information, where C1 depicts our probability plot, C2 a confusion matrix and C3 complements information displayed in the confusion matrix. In D1 the physiological data view shows the hypnograms for a selected patient. Alternatively, extra models can be visualized together with those the selected in A as shown in D2. Finally, D3 shows the signals that represent the input data.

now to allow for quick switching between distinct models. For instance, when exploring the hypnogram of a patient, it is interesting to switch between different models to verify if differences arise in terms of performance. In PerSleep, we compare two models: a *test* and a *reference* model. When both models are selected, all the relevant performance information will be displayed accordingly.

7.5.2 **Patient Data**

The *patient data* component consists of four views to provide an overview of the distribution of our entire population of patients and mechanisms to select and create groups of patients.

Patient Attribute Views

After discussing with the experts in sleep staging their needs, we opted for two linked views to present the patient's data in two ways: a barchart plot (Figure 7.3B1) and a parallel coordinate plot (PCP, Figure 7.3B2). Experts can decide what attributes to show at each time, focusing the analysis in specific parts of the entire dataset.

The barchart plot is useful to gain an understanding of the data distribution quantitatively (task **T1**). The PCP can be used to understand correlations in the data attributes (task **T2**). We use curves as a solution for the crossing problem [58]. The PCP provides a mechanism to have the same axis ranges for a set of selected attributes. This is useful to make direct comparisons between attributes (e.g., accuracy and kappa) when they have different value ranges. Experts can manually arrange the axes of the PCP in any desired order. PerSleep uses colors to depict patients belonging to groups of interest, i.e., created groups.

Some of the data attributes presented in our approach are computed dynamically when a model is selected. These attributes aim to summarize the information contained in the hypnogram and are meant to help detecting scenarios like those depicted in Figure 7.2:

Performance metric per third of the night This attribute can be used to inspect how the accuracy and kappa fluctuate during different parts of the night (see Figure 7.2a). Experts in the sleep domain usually divide the night into three equal parts to carry out their analysis (e.g., non-REM parasomnias usually occur in the first third of the night).

Pair-wise alignment metric The Smith-Waterman sequence alignment [152] is a local sequence alignment that we apply in order to verify how well test and reference model align. This helps to detect situations in which the accuracy metric is poor but the overall pattern of both hypnograms is somewhat similar (see Figure 7.2b).

Transitions and Sleep Fragmentation Index (SFI) The number of transitions as well as the SFI [112] can be useful to detect situations in which a model does not recognize sleep fragmentation adequately (see Figure 7.2c).

Sometimes, it is interesting to incorporate new data attributes. For instance, the expert may want to know how many transitions there are for a certain stage to verify a hypothesis generated during exploration. To this end, our approach enables the creation of these attributes

on the fly, which are then saved for subsequent exploratory sessions. To do so, users need to manually code how the values of these new attributes are computed by using the JavaScript language. A dialog can be opened by clicking on the cog icon present above the barchart plot and the PCP. This mechanism is meant to be used by users with knowledge on scripting languages, which is usually the case for ML experts.

The barchart and the PCP highly rely on brushing and linking to perform selections. When a selection is made in one of the views, the other is updated with the same selection. Having both perspectives (i.e., quantitative and correlative) linked helps the experts to better understand the global context of their data. More precisely, having visual feedback in the barchart when performing a selection in the PCP aids to understand whether the data selected follows a different distribution compared to the entire population or not.

Group Manipulation View

In Figure 7.3B3 the *group manipulation view* enables creation and manipulation of groups of patients. Created groups and their number of patients are shown in this view, which helps addressing **T1** and **T2** for groups of patients. Every group is identified with a name and a color that are assigned when created. In our system, we assume non-overlapping groups. Once patients have been added to a group, the PCP component updates accordingly, color coding each patient with the group color that it belongs to.

PerSleep provides a mechanism to create groups automatically using DBSCAN [43]. The clustering technique is fed with the normalized confusion matrices for each patient. Normalization is done *per patient* according to the total number of epochs. The aim of the clustering is to help experts in finding groups of patients that have similar model performance. The technique takes two parameters: *minPts* and ϵ. We set *minPts* to be 2 as we are interested in groups that contain at least two patients. ϵ can be adapted to enable exploration of different cluster outputs. We use an euclidean distance as the distance metric.

Groups can be selected on demand to explore performance data exclusively for those patients contained in the selected group.

Population View

The *population view* (see Figure 7.3B4) contains a table-like view that displays descriptive information of the recording of the patients and a summary of the disagreement for the selected models. We opt to explicitly encode the differences between the two sequences [54] by computing the epochs in which they disagree. This can be utilized to spot regions of disagreement, which partly supports task **T4**. Our visual encoding guarantees visibility [114] of the disagreements. Moreover, visual aids are added to depict three partitions of each sequence to ease comparison between patients with different sleep duration. Recordings can have different lenghts, which are shown numerically. For each patient, the visualization of the recording is stretched over the full width of the column to ease comparison. This is motivated by discussion with experts who needed to understand if errors present any pattern at a night

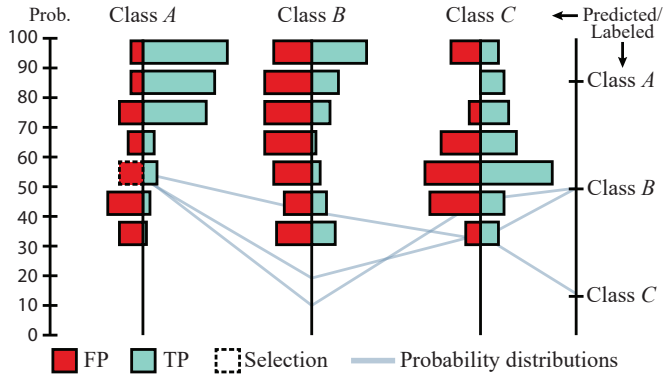


Figure 7.4: Glyph design of *probability view*. A multi-axis depicts the distribution of *true positives* and *false positives* per class. A line plot reveals the probability distribution for each prediction contained in a selected bin. The right axis depicts the labeled class.

level. The population view also indicates the groups to which patients belong. It is done by color coding the icon button placed on the left side of this view.

Users can sort the table of patients based on data attributes. This helps to quickly find patients for which a model performs the worst or the best. This view provides ways to select a patient for further exploration. Once a patient is selected, the patient data view, performance view and physiological data view change accordingly to show the information for this patient. We opt to keep the selected patient always visible by creating a visual duplicate on the top of the list. This is very helpful for experts to be aware of the patient that was selected, even when scrolling through the list. The *group view* updates to indicate the group to which this patient belongs by changing its visual encoding (see Figure 7.3B3). Moreover, patients can be unchecked such that they are not included in a group upon group creation. This gives a fine-grained method to exclude patients from the selections performed in the *patient attribute views*.

7.5.3 **Performance View**

To support task **T3** we introduce the performance view, which is composed of two components: *probability view* and *confusion matrix view*. They are aimed to be used together to gain insights into the performance of the model.

The *probability view* depicts a multi-axis view that conveys information about the probabilities for every epoch (see Figure 7.4). Each axis depicts the probability for each sleep stage, which are divided in ten bins of equal size ranging from 0 to 100. We distinguish between two categories: *agreements* and *disagreements*, which are placed on the right and left of each axis respectively. Each category features a vertical barchart depicting the number of predictions for a specific probability. The agreements represent the *true positives*, whereas the disagreements can represent either *false negatives* or *false positives* based on the choice of the

user. *False negatives* for class C in a multiclass problem are those samples where the *reference* model classified as C, whereas the *test* model classified otherwise. Similarly, *false positives* are samples where the *test* model classified as C but the *reference* model did differently.

This view is linked with other components and updates when a selection is performed to accordingly show the aggregated values. Also, the expert can select specific bins in this view to generate a line plot in the background (see Figure 7.3C1). It displays a double histogram for each class where each bin depicts the probability of a prediction. For each histogram, the right side shows agreements between both models (i.e., true positives and true negatives), whereas the left side depicts disagreements (i.e., false positives, false negatives or a combination of both). Moreover, the right-most axis depicts either the labeled or predicted class depending on whether the user is interested in *false positives* or *false negatives*, respectively. This complements the information shown in the *confusion matrix view* to give a better overview of the performance of the model.

The *confusion matrix view* (see Figure 7.3C2) provides a performance summary for every sleep stage. It is a standard way of presenting performance in multi-class scenarios. Although it can present information for any number of distinct classes, it is a good practice to keep this number low. In general, we deal with up to five distinct classes in sleep staging, thus clutter is not an issue. Previous work [52] also used a confusion matrix to guide the user to find potential misclassifications when ground truth is not available. In PerSleep, ground truth is available. Therefore, our confusion matrix shows the actual data rather than an estimate.

Sleep staging is, by definition, an imbalanced problem [163] with higher frequency of class *N2*. This poses the problem of getting biased insights when visually inspecting the confusion matrix. We provide an interactive confusion matrix where experts can decide how to display the data in the cells. Four possibilities are available: whole numbers, percentages, precision and recall. Percentages, precision and recall values are shown as ratios in our approach. When selecting recall or precision in the confusion matrix, the visual encoding adapts to better convey that numbers are *normalized* per column or per row. The confusion matrix can convey a general message on where accuracy and error occur more often. However, it is difficult to get an idea on the relative distributions of errors. For this reason, we accompany the confusion matrix with a double, vertical bar chart (Figure 7.3C3). The top side shows the *true positives* and *true negative* epochs (i.e., matching epochs), whereas the bottom side displays either *false negatives*, *false positives* or a combination of both (i.e., mismatching epochs).

Next to the confusion matrix, accuracy and kappa statistics are shown. These are understood by most experts in sleep staging.

Experts can interact with the cells of the confusion matrix. When one is selected, it filters out those patients that have at least one epoch in the selected category. For example, experts can select patients where *REM* is confused with *N3*. This particular situation is of interest since *REM* and *N3* are conceptually very different.

7.5.4 **Physiological Data View**

The *physiological data view* is shown in Figure 7.3D1, D2 and D3 and addresses task **T4**. This component provides a close look at the predictions of the test model, the ground truth of the reference model, and the signals for a single patient. Restricting to a single patient is motivated by discussions with the somnologist who stated that visualizing hypnograms simultaneously for many patients would rather obfuscate the analysis. By default, the hypnograms of the selected models are displayed. However, the expert may alternatively choose to visualize the hypnogram of other models (Figure 7.3D2) for quick comparisons.

This view features a piano-roll visualization for both test and reference scored hypnograms (see Figure 7.5a). It provides a visual overview on how similar they are. Both hypnograms follow a linear representation, with relative scale and unified layout [19]. Sometimes, deviations can be subtle and difficult to spot. To this end, we propose a combination of juxtaposition and explicit codification of differences [54]. The *difference view* follows the same principles as in the *population view*. The piano roll encodes sleep stages with position and color. Experts can switch to a single color piano-roll, which is closer to the representations they currently use in the sleep domain, or customize the colors for each sleep stage, which updates the entire interface of PerSleep. Our previous work [52] featured a similar view, however it was restricted to just one hypnogram to enable experts to spot and brush interesting patterns to find misclassifications.

When probability data is available, experts can choose to visualize it together with the hypnogram (see Figure 7.3D1). This provides an overview on how the probabilities fluctuate, potentially signaling situations in which the model was not sure about a prediction. Similar to our previous work [52], the probability is encoded in the background of the hypnogram. Two modes are provided: predicted sleep stage or chosen sleep stage. The first projects the probability of the class that is predicted in a certain epoch, whereas the second enables experts to visualize the probability of a sleep stage for all the epochs. This is useful to generate hypotheses about what the model detects in the input data.

Basic demographic information, such as age and sex, is shown in this view when a patient is selected. These demographics are important for experts in order to correctly interpret the hypnogram. The sleep of young and old people is different, for example. Patient ID and file name are shown for completeness such that the expert can quickly identify the patient being visualized and from which file the PSG is taken. Experts can explore patient data in full detail by clicking on the medical notes icon, which opens a new window where the full data is listed in tabular form.

A widget on the upper left brings the set of signals. The expert can select any available signal to be shown below the hypnograms (Figure 7.3D3). This provides a flexible mechanism to deal with models that have different input signals (e.g., with and without respiratory information). Every signal is shown through a time window, which can be configured by the expert. Navigation is enabled in *epoch* units. The expert can set the length of the signal in seconds to be displayed, giving more flexibility to explore the epochs surrounding a prediction.

Two main interactions are provided. The first one concerns selections of parts of the hypno-

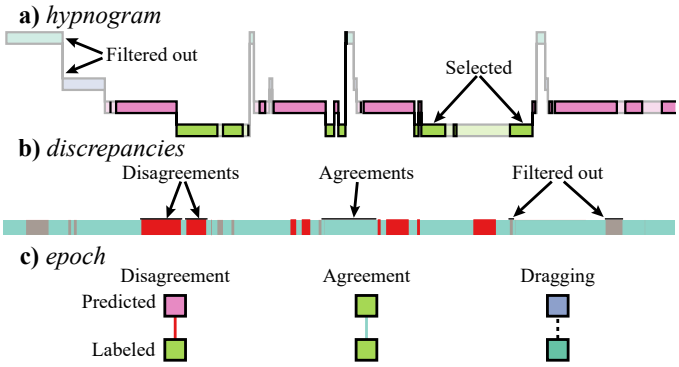


Figure 7.5: Glyph design used in *physiological data view*. a) depicts the way we visualize hypnograms when selections are done in the *performance view*; b) shows the way we encode the differences between predicted and labeled hypnograms, where the colored stripes indicate disagreement and different opacity is used to depict current selection; c) presents the slider used to select a specific epoch.

grams with focus+context. To this end, we rely on brushing over the *difference view*. It facilitates the selection of disagreeing, or interesting in general, fragments. The second interaction enables quick navigation to the input data for a specific epoch. This can be done by dragging a slider over the hypnograms. The design of the slider provides hints indicating the class of the test and the reference models. A line connecting both ends encodes whether they both classes agree. Examples can be seen in Figure 7.5c.

7.5.5 **Complexity and Scalability**

PerSleep has been implemented as a web app that entirely runs on *client* side. This means that data never goes out of the local machine of the user, and everything remains in the local storage of the web-client. This decision was made to ensure there are no privacy issues with the data being analyzed. However, it also introduces limitations that need to be addressed.

The visualization of the input data besides the outcomes is important for performance evaluation. In sleep staging, many physiological signals are recorded, resulting in a large amount of data. In order to enable a smooth data exploration, we make use of WebAssembly [61] to run EDFlib, a C library that reads EDF [80] files efficiently. With this approach, we achieve a nearly native speed, resulting in a smooth and viable data exploration.

Another consideration goes for the computation of data attributes. As discussed in previous sections, our approach provides mechanisms to define new attributes. These are recomputed on demand every time a new model has been imported. This ensures that PerSleep remains responsive during the exploratory phase.

We have run experiments with 236 patients and 463,090 epochs. In other experiments based on real-world data, we handled smaller amounts of data in terms of patients and, thus, number of epochs. Our approach has been able to handle all our experiments adequately and no concerns arose from our users. We have observed that the *probability view* tends to be

computationally most demanding of our system. However, we have not experienced any significant impact on the interactive performance for real-world use cases.

7.6 Use Case

We demonstrate our approach on a real-world multivariate sleep dataset which contains three different alternative sleep staging models, based on the use of ECG, respiratory effort and ECG+respiratory data. The data that these models consume can be extracted from surrogate devices. For example, the ECG model is fed with heart rate variability, which can be measured with most modern smartwatches. The interest of the experts in these models is to verify if they could replace the gold standard (i.e., PSG) to perform sleep studies in a less intrusive manner. The three models output 4 classes: *awake*, *N1+N2*, *N3* and *REM*. The dataset was recorded in a Dutch hospital as part of a study about sleep in 74 patients with intellectual disabilities more than 16 years old who suffered from a variety of sleep problems. Some patients present comorbidities such as heart problems and epilepsy. The patients received a PSG during routine clinical care. Patients with absent ECG channels or poor ECG and EEG were discarded from the study. Patients' attributes such as age, sex, comorbidities, primary diagnosis, whether they receive medication, etc. are available in the dataset. We show how the visualizations and interactions in our approach help experts to gain insights into this dataset. The use case was performed throughout several interactive group sessions with a somnologist, a machine learning expert and a signal processing expert. The three users are all knowledgeable about sleep and machine learning. A multidisciplinary setting is common for assessing the performance of a ML model. Each expert can add a different point of view: the somnologist provides a clinical perspective, the ML expert adds knowledge about the model and the signal processing expert includes feedback about the signals fed to the ML model.

Generally, interesting patients are those that exhibit an extremely good or bad performance compared to others. Initially, we created a case study in our system containing the patients recorded in the previously defined study. Then, we incorporated the data from the three models, resulting in 79,573 epochs per model.

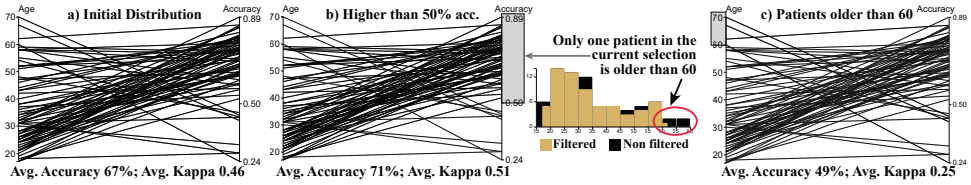


Figure 7.6: a) initial set of patients for ECG model; b) selection with accuracy higher than 50%; c) selection of patients older than 60.

First, experts selected the ECG model as test model, and human scoring based on EEG as the reference model. We started the exploration by inspecting relationships between data attributes. Experts wanted to verify if there was some correlation between basic demographics and accuracy. For this, the experts selected age in the barchart view, and age and accuracy

in the PCP. After a first inspection of the PCP, the experts observed a certain trend of lower accuracy scores for older patients. In particular, for 3 out of 4 patients (75%) that are 60 or older, the ECG model scored lower than 50% in terms of accuracy. This contrasts to patients that are younger than 60, where for just 6 out of 70 of them (9%) the model scored lower than 50% accuracy (Figure 7.6). This finding was interesting, as there is no data on age dependency of alternative sleep staging models. We explored the same set of patients with the respiratory model. In this case, only 1 out of 4 patients scored lower than 50% accuracy, which may indicate that the ECG model is not reliable for older patients. A closer inspection of this patient revealed that the ECG model was not able to detect any *REM* stage, and most of times the model confused *N3* with *N1/N2*.

Epilepsy appears to be more prevalent among people with intellectual disabilities. In fact, it is believed that up to one-fifth of the population with intellectual disabilities also suffer from epilepsy. Epilepsy certainly has an impact on sleep abnormalities. In particular, it can increase sleep latency (i.e., time to fall asleep), sleep fragmentation, awakenings and stage shifts [104]. However, in addition, epileptic activity in the EEG can make it more difficult to annotate sleep stages. For these reasons, the experts wanted to understand how the ECG model was performing on these patients. To this end, the experts created two groups: patients with and without epilepsy. These groups resulted in 28 and 46 patients respectively. We observed that average accuracy and kappa values were very similar for both groups, which suggests that the ECG model is not performing differently for patients with epilepsy.

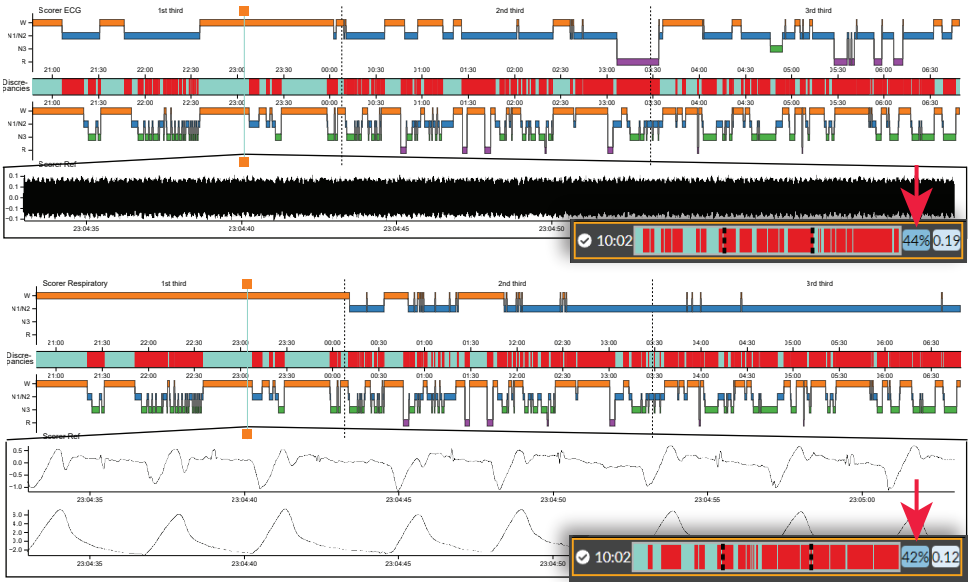


Figure 7.7: Patient (ID 51) with diagnosed epilepsy. The ECG model (top row) performed slightly better than respiratory model (bottom row) although the former used data with many artifacts as input, which is shown for epoch 275 and a 30 seconds window.

To gain more insights into the epilepsy patients, the experts sorted the patient list by accuracy to focus on the top and bottom cases. In particular, we found one case with low accuracy

(44%) which contained only ECG artifacts (Figure 7.7 top). Despite this, the model was still able to classify some sleep stages. The experts switched the test model to check how the respiratory and ECG+respiratory models performed. The respiratory model, whose data does not seem to contain artifacts, scored slightly lower (42%) than the ECG (Figure 7.7 bottom). However, it was interesting to note that the combination of both (ECG+respiratory) provided a slightly better performance (48%). Experts looked at comments made by the sleep technicians by clicking on the medical notes icon. It was clear that the technician scoring the reference sleep recording had trouble identifying some sleep stages while it was easy to distinguish between *wake* and *sleep*. It was difficult to decide between *N1/N2* and *N3* in this patient. After reading these annotations, the experts suggested that the ECG model may be providing a better scoring than the human EEG based scoring.

From this moment of the analysis, experts focused on selecting patients from the full patient list to obtain an idea on common problems. Experts detected two common situations: *REM* misclassifications and *N3* fragmentation. Regarding *REM* misclassifications, the experts proposed that this could be caused by sleep apnea (often occurring in this stage), or autonomic dysfunction leading to abnormal ECG patterns in *REM*. By inspecting the data, they were able to verify that a breathing disorder was indeed diagnosed for some of these patients.

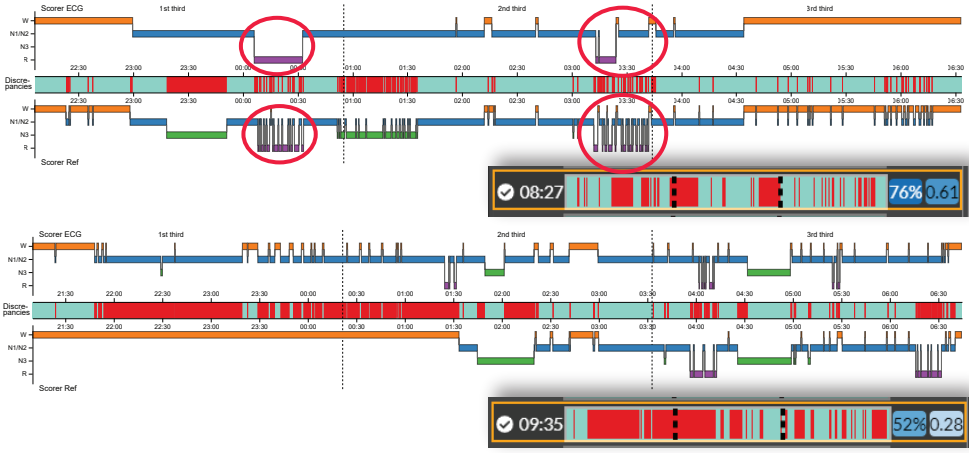


Figure 7.8: Examples of misleading statistics. The top row (ID 114) presents a case in which the test ECG model does not recognize *REM* fragmentation (see red circles). Bottom row (ID 133) have similar clinical interpretation (i.e., similar overall pattern) despite the low accuracy and kappa of the test model.

During this part of the analysis, experts found cases in which aggregated metrics (accuracy and kappa) were absolutely misleading. In particular, they found cases where kappa and accuracy were high, but the clinical interpretation of the two hypnograms would be very different (Figure 7.8 top) due to the absence of sleep fragmentation. Similarly, we found some cases in which kappa and accuracy were low, but the clinical interpretation of the test model would most likely be the same as the reference model (Figure 7.8 bottom) because the overall pattern of transitions looks alike for the test and the reference model.

7.6.1 **Sleep Fragmentation**

Sleep fragmentation is one of the problems depicted in Figure 7.2. One of the causes of sleep fragmentation is medication. To analyze how our model copes with this, the experts selected three attributes in the PCP view: transitions in the reference model, medication and transitions in the test model. Immediately, they observed two things: medication seems to have an influence in the total number of transitions in the reference model but not in the test model; and the test model produced a significantly lower number of transitions for all the patients (see Figure 7.9). The latter may indicate that the model is *smoothing* the resulting hypnograms.

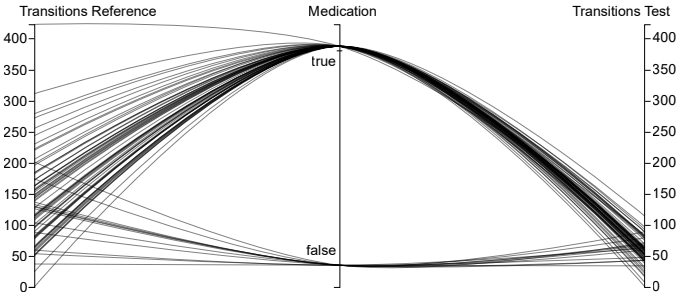


Figure 7.9: PCP comparing transitions in test and reference model against medication. As can be seen, the test model seems to smooth the number of transitions for patients with and without medication.

In order to verify this hypothesis, the experts created two groups of patients. The first group contained patients with a low number of transitions (≤ 150) whereas the second had a high number of transitions (> 150) according to the reference model. Initially, they noticed that both groups had different performance metrics (70% vs 60% accuracy; 0.5 vs 0.4 kappa). Next, they wanted to gain more insights into the behavior of the model for each group. They started the analysis by selecting the group with many transitions and sorting the patient list by accuracy. After inspecting several patients with different accuracy values, they discovered that sleep fragmentation was indeed not correctly detected in the vast majority of the cases. They repeated the process for the other group of patients. They found out that the majority of patients did not present a very fragmented hypnogram, but they found a few cases that presented fragmentation and yet the model produced a smoother version that omitted such fragmentation. This may indicate that the model is not able to capture the transitions between sleep stages as a human scorer would.

7.7 **Discussion**

The use case presented in Section 7.6 shows how PerSleep can be used for an exploratory analysis to evaluate the performance of a ML model for sleep staging. This is done by leveraging the proposed design to explore the data from different perspectives. Tight linking between patient data and epoch data provides the somnologist and the ML experts a mechanism to generate and test hypotheses that would otherwise require a lot of manual work.

The definition of performance for sleep staging is ill-defined due to the coarse-grained nature of traditional metrics that are often provided when evaluating sleep staging classifiers. According to the somnologist and ML expert: *"There was a clinical message out of this. It shows that the clinical interpretability of a surrogate or target hypnogram definitely is not fully determined by the kappa and accuracy numbers"*. This claim strongly supports the need for visual analytics systems that help in performing exploratory analysis of performance in sleep staging.

The current workflow for performance analysis in sleep staging is rather cumbersome. It often involves several steps performed in different platforms and software. Among the steps, we can find visual tasks like analysis of hypnograms, etc. Our approach unifies all the steps and provides mechanisms to link elements in a visual manner. A remark from the somnologist after the use case was: *"In papers, you very often look either at the group level or some illustrated hypnograms. One of the merits of PerSleep resides in the ability to analyze very quickly individual recordings (i.e., hypnograms) so one could search for specific reasons why discrepancies happen, which can really be different from subject to subject"*.

In general, working with clinicians limits the choices when designing a visual analytics system. For example, we found that the PCP was on the edge of complexity for the doctor. This motivates using simple visualizations. Clinicians are used to some graphical representations (in this case hypnograms). These traditions must be respected and included in the final design. Otherwise, the system may become unusable for clinicians.

As for other approaches, ours has limitations. For instance, ours does not deal with techniques that use inputs such as images. This affects the generalization of our approach. Also, we heavily rely on prior knowledge from the expert on the models being analyzed such as the input data, or the set of patients (e.g., healthy or not). Additionally, the creation of new data attributes relies on prior knowledge with scripting languages, which can be a problem for experts on sleep medicine that lack of a more technical background. Finally, our approach does not provide mechanisms to distinguish between findings that may not be statistically significant. We however believe this is alleviated because the users can rely on their expertise to decide whether the findings are representative or not.

7.7.1 **Approach Generalization**

Our approach can be generalized to other domains. In the medical field, it can be applied in epilepsy prediction [147]. In this context, ML models are used to detect seizures from EEG data. Hence, this domain shares many similarities with sleep staging: predictions are collected per patient, which also has multivariate data, and they are sequential. The data abstractions of the epilepsy and the sleep staging domains are alike.

In epilepsy detection, ML models are trained to, at least, detect two different classes: *seizure*, and *non-seizures*. These classes can be split into more specialized ones to characterize the severity of the epilepsy: *simple-partial*, *complex-partial*, *generalize convulsive* and *generalize non-convulsive* seizures. The former two epilepsy seizures happen in one hemisphere of the brain, whereas the latter two happen in the whole brain. Usually, the duration of a seizure

ranges from seconds to minutes. Users in this field would be interested in correctly recognizing the type of seizures to determine the severity of the epilepsy attack and correlate the predictions of the model with other clinical data. Our approach was designed to address the tasks stated in Section 7.3. Except **T4**, which involves comparing hypnograms, the remaining tasks would still be suitable in this field. Additionally, locating the origin of the seizure is important in epilepsy detection. In this regard, the EEG (i.e., the brain electrodes) already gives some clues on where the seizure took place. Therefore, it would be necessary to add a new task. This would help in locating the origin of the seizure.

7.8 Conclusions

We presented a novel approach to evaluate the performance of ML models for sleep staging. It combines different visual and interactive components to enable experts to conduct their exploratory analysis. In contrast to related work, we address a problem that involves predictions over time in combination with patient data, which cannot be dealt with using current approaches. A use case has been presented to demonstrate our approach where we describe the main discoveries made by experts during exploration.

In principle, a similar approach like ours can be used to any situation where complex dynamic signals have to be judged for the state of the object of interest. In our case, we took care to understand the needs of our collaborators and carefully tuned the system accordingly. This may be simply the way to go, but it is also intensive. The design of a generic, flexible system that enables similar functionality without programming is still an open challenge.

8

Conclusions

In the previous chapters of this dissertation we attempted to answer the research question introduced in chapter 1: "How can we use interactive visualization and automated techniques to understand and optimize medical workflows?" This chapter discuss the main conclusions of our work and directions for future research.

8.1 Achievements

In this dissertation, several visualizations, interactions and experiments were presented. They all cover the broad topic of improving medical workflows and target specific topics within process mining and machine learning. The presented approaches, which are in form of interfaces, prototypes and use cases, helped to answer the research question. Our research was divided into two main areas, namely process mining and machine learning. An overview of the conclusions of chapters 3 to 7 is presented below.

Soundness Analysis in Petri nets
In chapter 3, we presented a tool to visually assess the soundness of a Petri net, where analysis tasks were presented. We demonstrated the usability of our approach with two use cases. One limitation of our approach is that it relies on the state space, which cannot always be computed in a reasonable time.

Performance and Conformance Checking for Process Models
In chapter 4, we proposed an approach to analyze process-centric information of an event log in combination with clinical longitudinal data. We demonstrated our approach on an sepsis dataset. We showed that our approach can be used to assess conformance between modeled and real behavior of a clinical process.

Interactive Correction of Deep Learning Predictions in Sleep Staging
In chapter 5, we presented a visual analytics approach for interactive correction of machine learning predictions in the real world. It is a novel visual analytics approach applied in a sleep staging context. The provided use case shows that our approach can be used to find and correct suspicious predictions. Discussion with experts revealed their genuine interest and also opportunities for improvement. Although a perfect correction is not guaranteed, our approach does enable experts to analyze interesting or suspicious patterns.

Explainability for Sleep Staging
In chapter 6, we introduced the problem of making sense when dealing with temporal inputs in deep learning approaches. Sleep staging is an example of scenario where temporal inputs are used to score the sleep of a person. In our experiments we found that the model seemingly detected patterns in the input data. However, such patterns were not easy to relate to medical concepts such as k-complexes, spindles and alpha waves. Our experiments focused on a single channel input, which can be a limitation for models trained on multiple channel.

Performance Assessment of Sleep Staging Models
In chapter 7, we presented a novel approach to evaluate the performance of ML models for sleep staging. In contrast to related work, we address a problem that involves predictions over time in combination with patient data. Our approach enables for a flexible exploration that takes the patient data and the performance at its core. A use case was presented to demonstrate our approach, which showed that experts were enabled to make useful discoveries during exploration.

8.2 **Research Question**

The previous chapters of this dissertation presented visual methods and experiments in an attempt to answer the research question proposed in chapter 1:

How can we use interactive visualization and automated techniques to understand and optimize medical workflows?

Moreover, in the same chapter, we divided the research question into smaller, more targeted questions. Figure 8.1 depicts an overview of the main contributions presented in this dissertation and the relations between these, which helped to answer the research question. Moreover, a discussion on how our proposals contributed to answering these questions is presented below:

1. *How can we understand the circumstances under which soundness breaks down in a Petri net?* The visualization and interactions proposed in chapter 3 provide insights into soundness verification for process models. The method used in our approach [181] to compute the state space employs a threshold to decrease the depth of the space to be explored. Although this workaround simplifies the computation of the state space, it can still take considerable time to finish if the semantics of the Petri net are too complex. Alternatively, an incremental approach could be used to alleviate this issue. In this sense, progressive visual analytics [159] could be a perfect match for this task.

   In Petri nets, we have places and transitions. Places can be contained in multiple states of the state space. This was one of the challenges we addressed with our design as this information is important to have a better context of the role of a place in the semantics of the Petri net. For example, we can visualize if a place belongs to a single final state, multiple or none just with an overview. Regarding soundness, we use a combination of multiple views to show three important aspects for soundness verification: the Petri net, interesting states of the state space and the runs. These views and the interactions between them enable the exploration of interesting states.

   Our approach showed to be effective in analyzing the circumstances under which soundness breaks. It represents a step ahead in comparison with traditional techniques where soundness verification is not presented together with the process model, making it more difficult to understand the circumstances under which the process is no longer sound. Our approach addresses this gap and enables experts to analyze these circumstances.

2. *How can we gain insights into the modeled and real processes of hospitals with conformance checking?* Methods proposed in the literature to assess the conformance of a process in a hospital did not consider the patient perspective. Linking both process and clinical data can result in valuable insights into the processes running at hospitals and the guidelines they follow to treat certain diseases. Taking the patient perspective into account helps to better understand the process followed in a hospital. For example, deviations can occur based on the result of lab tests, delays in the delivery of drugs

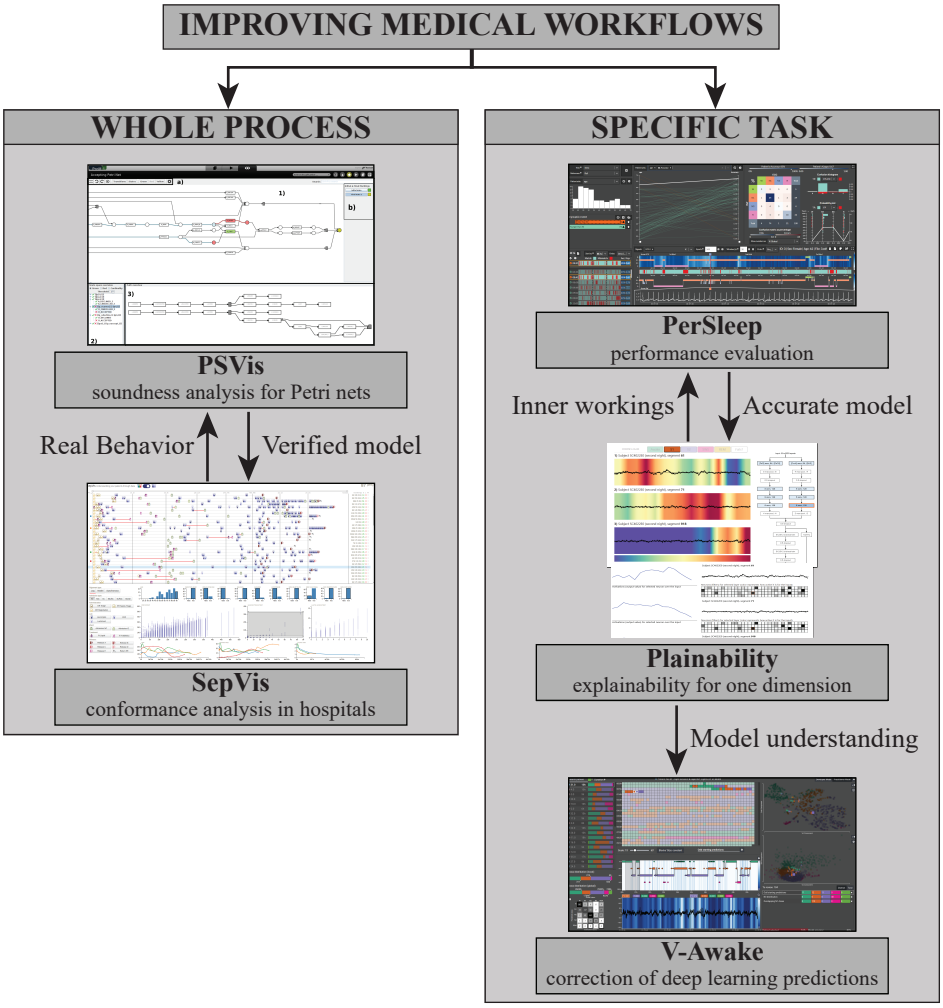


Figure 8.1: Overview of the methods and experiments described in this dissertation and the main interactions between them.

and problems in the administration of the hospital. The patient data perspective together with process mining analysis techniques is the major novelty of our approach. Patient information is not currently taken into account by automated techniques due to the complexity of formalizing all possible cases and guidelines. Besides, doctors rely on their own experience to make decisions, which can make the incorporation of such knowledge in an automated approach infeasible. Our design can also be used to find ways of dealing with certain diseases that do not match the guidelines. This can be useful to explore new ways of treating certain patients or to detect problems with guidelines.

We found that having domain knowledge is important to operate a system like ours. Chapter 4 presented a sepsis case. The guidelines to treat this disease impose certain time frames to run tests and provide medications. Such domain knowledge is fundamental to explore how the real behavior fits those time constraints. Our design enables experts to filter cases based on different constraints so they can replicate those imposed by the guidelines and verify whether they are fulfilled. This flexibility is crucial for the generalization of our solution, as this can be applied to other domains with minor adaptations.

3. *How can we support the usage of machine learning in a real-world, clinical setting?* Machine learning represents a promise for the future of healthcare. The usage of automated models can be problematic due to the lack of trust and confidence in the output of such models, making adoption in a real world setting even harder. Our proposal involves several views on the same data that can help experts to find potential misclassifications. Our goal is twofold: fixing misclassifications and building trust.

   Despite the importance of the problem tackled, no other techniques have yet been proposed in the same direction for sleep staging. With V-Awake we found out that the combination of dimensionality reduction techniques, detailed views of the epochs, expert knowledge and interaction was sufficient to detect potential misclassifications. We also realized that patients with abnormal sleep behavior may be difficult to analyze since abnormal patterns may be wrongly taken as misclassifications.

   Acquiring extra knowledge about the inner works of the model is beneficial to enable a more efficient exploration. To this end, chapter 6 showed experiments with a neural network. In these experiments, we found that some layers of the model were recognizing intuitive patterns in the input data. This can be used by experts to validate the features extracted by the model to make certain classifications, which enhances the trust in the model.

4. *How can we evaluate the performance of a machine learning model for different groups of patients?* Performance assessment in machine learning classification is challenging by itself. Generally, just an overview or a high-level snapshot of the performance per class is examined. In clinical settings, however, the performance of the model may be influenced by other factors that are not evident with a traditional performance evaluation. Naturally, each person has different underlying physiological characteristics. This is also the case in sleep, where for instance the sleep of a young person differs from the sleep of an old one. In order to enable machine learning experts to better evaluate the performance of models for sleep staging, we combined performance information (e.g., predictive accuracy, Cohen's kappa value, etc.) with patient information (e.g., sex, age, medication, previous disorders, etc.) in chapter 7. The usage of multiple linked views has the advantage of exploring the data from several perspectives. Moreover, the creation of subgroups of patients was beneficial to divide the cases to be explored in smaller and more comprehensible groups. We discovered that easing the inspection of hypnograms in conjunction with patient data is effective to better understand the performance of machine learning models in sleep staging.

   One of the biggest challenges to evaluate the performance in sleep staging is that mul-

tiple problems can arise simultaneously. Chapter 7 showed three common problems, although others exist. Therefore, our design needed to address multiple situations in a flexible manner. We incorporated a PCP to explore correlations in data attributes. Moreover, the ability to compute new data attributes proved to be effective to enable a flexible exploration of the data and quickly test new hypotheses.

8.3 Directions for Future Research

The methods and experiments presented throughout this dissertation, described in chapters 3 to 7, enabled us to make advancements in improving medical workflows. We validated our prototype solutions by presenting use cases and discussing our work with domain experts. In the process of our research, we could not tackle all problems. For this reason, in this section we provide pointers for future research.

Applicability
The methods presented in chapters 3 to 7 were applied to specific domains such as Petri nets, sepsis and sleep staging. This limits the applicability of the presented approaches. However, our methods are generic enough to be applied to other applications in healthcare. Chapters 5 and 7 present methods applied to sleep staging, but they also provide directions for application in other domains where time series data plays an important role, such as epilepsy detection and colon cancer video inspection. The method presented in chapter 3 concerns Petri nets, which can be a limitation for the applicability of our method. Many process models can be converted into Petri nets, which can help dodge this limitation. Finally, chapter 4 presented a combination of process mining and visualization techniques applied for a sepsis case. Researching how to adapt our methods for bigger process models with hundreds of activities would be crucial to apply our methods in such scenarios.

Machine Learning in the Real World
In chapter 5 we assumed the model to be static, but the integration of active learning may be greatly beneficial in the design presented in chapter 5. It could be used by experts to improve the model when potential misclassifications have been found. This requires more research to ensure that the learning process provides benefits. For example, it may be the case that our model loses the ability to generalize, resulting in overfitting. Furthermore, the dimensionality reduction plot presented in chapter 5 can be raised to a higher level. In particular, focusing on the entire cohort population rather than a single patient could provide some further insight into patients that have more faulty predictions. Our intuition is that we could apply a dimensionality reduction method over the entire population to find groups of patients that share similar peculiarities in terms of activations of layers. With this, users would only have to analyze some representative subjects from a particular cluster and apply the learned facts to the rest (e.g., majority of faulty predictions in stage *REM*, similar sleep patterns, etc.).

Process Mining and Patient Data
Future directions should enable the user to work in terms of cohorts of patients and present correlations, distributions and process deviations between them. Moreover, research on how

to increase the scalability of the design of chapter 4 to deal with larger and more complex processes is necessary.

Chapter 3 used automated techniques to compute the state space of a Petri net. Future directions must consider alternatives to compute just the parts of the state space that are used in the analysis. One option might be to explore the state space incrementally by computing just portions and incrementally compute the rest on demand with user interaction. In this context, progressive visual analytics [159] may be a good fit. Furthermore, displaying the runs in an intuitive matter is not a trivial task. The design presented in chapter 3 lacks an explicit way to display loops and concurrency, which would help to perform important tasks (e.g., examine concurrency, loops and causal order in runs). Experiments to compare different approaches for displaying the runs would be beneficial to discern the most suitable approach.

Machine Learning Performance and Patient Data
Clustering can be beneficial in performance evaluation to find groups of patients that share similar problems. This follows the same reasoning as for the future directions presented for machine learning in real world. Future work should extend the clustering capabilities of the approach described in chapter 7 and evaluate different techniques and distance metrics. As proposed by the experts, analyzing more than two classifiers simultaneously can ease the task of performance assessment. However, it is not entirely clear how to combine all this information in a visual analytics approach that remains simple enough to be used by domain experts. Finally, analyzing the uncertainty of a model would be greatly interesting. A motivation example is the work of Stephansen et al. [158], where the uncertainty produced by a sleep staging classifier was used to model a narcolepsy classifier, raising expectations for the discovery of other sleep-related disease markers. Incorporating this information in a visual analytics system could help to discover new markers.

8.4 Lessons Learned

Throughout the development of this dissertation, I learned several lessons that I would like to pass to future PhD students in this area. I collected these in the following points.

Expert knowledge is crucial, but it also has limitations All projects presented in this dissertation contain, in a higher or lesser degree, expert knowledge. Usually, it steered the definition of the problems that we tackled and the final solutions. For this reason, expert knowledge is absolutely important to the success of a project. However, expert knowledge by itself is not enough to successfully deliver a good visualization design. Experts tend to stick to techniques and methods that are familiar to them. This may limit the scope of the choices and decisions taken in a visualization design. Finding a good balance between expert knowledge and visualization principles is the key.

Intuition can play in your favor Sometimes, intuition is somewhat disregarded in favor of a reasoning process. It is important to reason about every choice made in a visualization design, but intuition and inspiration are important as well. In particular, at the

beginning of exploring a new problem, where things are not absolutely clear, following your instinct can entail the success of such idea.

Data is often difficult to obtain When collaborating with other institutions, using their data for analysis is crucial as this represents real-world data. However, in some occasions, problems may arise with privacy and confidentiality, leading to delays and sometimes making access to that data hard, if impossible. Having a backup plan is important to ensure the continuity of your work.

Do not be intimidated The visualization community is continuously growing and there are very smart and experienced people. Sometimes, it may be scaring to present or defend your work in front of them, but it is absolutely worth it. The community is open and welcoming, and even the brightest people started as a student. Not being *the best* is allowed as long as you always aim to *your best*.

The medical domain is special Working with people involved in the medical field can be challenging. People in this domain often have to follow guidelines, privacy rules, and are under continuous time pressure. Moreover, their visualization knowledge is based on simpler techniques such as bar charts, line charts, etc., and they are only willing to spend time if it provides clear benefit. Carefully taking these constraints into account is vital for the success of a project.

8.5 **Finally**

A central topic in this dissertation is the applicability of machine learning models in real-world contexts. In particular, we focus on healthcare, which can greatly benefit from using machine learning in their daily workflows. The benefits are clear to everyone: improve time, reduce costs, assist doctors in taking difficult decisions, to name a few. However, many times we tend to avoid some obvious consequences. If machine learning models are not perfect, can we trust them to influence the decisions about real patients? This question significantly influenced some of the ideas presented in this dissertation.

Much work is done currently in better understanding complex machine learning models. This certainly helps doctors utilize these because they better understand how the model works. However, this may not be sufficient in the (near) future. With an increase in the usage of machine learning, there will be a more urgent need for systems that increase the trust of doctors. Visualization presents itself as a splendid candidate to this end. An indication can be seen in the increase of publications in the field of explainable artificial intelligence.

I absolutely believe that visualization and machine learning will play a key role in the future of medicine. This is the reason why I include these words at the end of my dissertation to encourage people in the field to work towards more trustworthy and reliable techniques that can help clinicians in real-world scenarios where ground truth is no longer available.

References

[1] **Aalst, W.** Verification of workflow nets. In *Application and Theory of Petri Nets 1997* (Toulouse, France, June 1997), P. Azéma and G. Balbo, Eds., vol. 1248 of *lncs*, springer, pp. 407–426.

[2] **Aalst, W.** The application of Petri nets to workflow management. *The Journal of Circuits, Systems and Computers 8*, 1 (1998), 21–66.

[3] **Adam, N., Atluri, V., and Huang, W.** Modeling and analysis of workflows using Petri nets. *Journal of Intelligent Information Systems 10*, 2 (Mar. 1998), 131–158.

[4] **Agarwal, R., and Gotman, J.** Computer-assisted sleep staging. *IEEE Transactions on Biomedical Engineering 48*, 12 (2001), 1412–1423.

[5] **Albarqouni, S., Baur, C., Achilles, F., Belagiannis, V., Demirci, S., and Navab, N.** Aggnet: deep learning from crowds for mitosis detection in breast cancer histology images. *IEEE transactions on medical imaging 35*, 5 (2016), 1313–1321.

[6] **Alturki, F. A., AlSharabi, K., Abdurraqeeb, A. M., and Aljalal, M.** Eeg signal analysis for diagnosing neurological disorders using discrete wavelet transform and intelligent techniques. *Sensors 20*, 9 (2020), 2505.

[7] **Amershi, S., Chickering, M., Drucker, S. M., Lee, B., Simard, P., and Suh, J.** Modeltracker: Redesigning performance analysis tools for machine learning. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (2015), ACM, pp. 337–346.

[8] **Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., and Chen, G.** Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning* (2016), pp. 173–182.

[9] **Anscombe, F. J.** Graphs in statistical analysis. *The american statistician 27*, 1 (1973), 17–21.

[10] **Balk, R. A.** Systemic inflammatory response syndrome (SIRS): where did it come from and is it still relevant today? *Virulence 5*, 1 (jan 2014), 20–6.

[11] **BERG**. Back to biology for a healthier tomorrow. `https://www.enlitic.com/`. Accessed on April 2021.

[12] **Bergenthum, R., Desel, J., Juhás, G., and Lorenz, R.** *Can I Execute My Scenario in Your Net? VipTool Tells You!* Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 381–390.

[13] **Berger, H.** Über das elektroenkephalogramm des menschen. *Archiv für psychiatrie und nervenkrankheiten 87*, 1 (1929), 527–570.

[14] **Bhardwaj, R., Nambiar, A. R., and Dutta, D.** A study of machine learning in healthcare. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)* (2017), vol. 2, IEEE, pp. 236–241.

[15] **Biswal, S., Kulas, J., Sun, H., Goparaju, B., Westover, M. B., Bianchi, M. T., and Sun, J.** Sleepnet: Automated sleep staging system via deep learning. *arXiv preprint arXiv:1707.08262* (2017).

[16] **Borland, D., West, V. L., and Hammond, W. E.** Multivariate Visualization of Longitudinal Clinical Data. *Proceedings 2016 IEEE VIS Workshop on Visual Analytics in Healthcare, Chicago,IL* (2016).

[17] **Bose, R. J. C., and van der Aalst, W. M.** Analysis of patient treatment procedures. In *Business Process Management Workshops (1)* (2011), vol. 99, pp. 165–166.

[18] **Bostock, M., Ogievetsky, V., and Heer, J.** D^3 data-driven documents. *IEEE transactions on visualization and computer graphics 17*, 12 (2011), 2301–2309.

[19] **Brehmer, M., Lee, B., Bach, B., Riche, N. H., and Munzner, T.** Timelines revisited: A design space and considerations for expressive storytelling. *IEEE transactions on visualization and computer graphics 23*, 9 (2016), 2151–2164.

[20] **Campbell, S. S., and Tobler, I.** Animal sleep: a review of sleep duration across phylogeny. *Neuroscience & Biobehavioral Reviews 8*, 3 (1984), 269–300.

[21] **Card, S., Mackinlay, J., and Shneiderman, B.** Information visualization. *Human-computer interaction: Design issues, solutions, and applications 181* (2009).

[22] **Care, S. H.** Shah lab. `https://shahlab.stanford.edu/`. Accessed on April 2021.

[23] **Celonis**. `https://www.celonis.com/`. Accessed on April 2021.

[24] **Chen, X., and Lawrence Zitnick, C.** Mind's eye: A recurrent visual representation for image caption generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 2422–2431.

[25] **Choi, K., Fazekas, G., Sandler, M., and Cho, K.** Convolutional recurrent neural networks for music classification. In *Acoustics, Speech and Signal Processing (ICASSP)* (2017), IEEE, pp. 2392–2396.

[26] **Chriskos, P., Frantzidis, C. A., Gkivogkli, P. T., Bamidis, P. D., and Kourtidou-Papadeli, C.** Automatic sleep staging employing convolutional neural networks and cortical connectivity images. *IEEE transactions on neural networks and learning systems 31*, 1 (2019), 113–123.

[27] **Chriskos, P., Frantzidis, C. A., Nday, C. M., Gkivogkli, P. T., Bamidis, P. D., and Kourtidou-Papadeli, C.** A review on current trends in automatic sleep staging through bio-signal recordings and future challenges. *Sleep Medicine Reviews 55* (2021), 101377.

[28] **Combrisson, E., Vallat, R., Eichenlaub, J.-B., O'Reilly, C., Lajnef, T., Guillot, A., Ruby, P. M., and Jerbi, K.** Sleep: an open-source python software for visualization, analysis, and staging of sleep data. *Frontiers in neuroinformatics 11* (2017), 60.

[29] **Cook, K. A., and Thomas, J. J.** *Illuminating the path: The research and development agenda for visual analytics*. IEEE Computer Society Press, 2005.

[30] **Corvò, A., Garcia Caballero, H. S., and Westenberg, M. A.** Survivis: Visual analytics for interactive survival analysis. In *10th International EuroVis Workshop on Visual Analytics, EuroVA@EuroVis 2019, June 3, 2019, Porto, Portugal* (2019), Eurographics Association, pp. 73–77.

[31] **Corvò, A., Garcia Caballero, H. S., Westenberg, M. A., van Driel, M. A., and van Wijk, J.** Visual analytics for hypothesis-driven exploration in computational pathology. *IEEE Transactions on Visualization and Computer Graphics* (2020), 1–1. Early access.

[32] **C.Plaisant, R. Mushlin, A. S.** LifeLines: Using Visualization to Enhance Navigation and Analysis of Patient Records. *American Medical Informatic Association Annual Fall Symposium* (1998), 9–11.

[33] **Cruz-Roa, A. A., Ovalle, J. E. A., Madabhushi, A., and Osorio, F. A. G.** A deep learning architecture for image representation, visual interpretability and automated basal-cell carcinoma cancer detection. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2013), Springer, pp. 403–410.

[34] **Dehnert, J., and Rittgen, P.** Relaxed Soundness of Business Processes. In *Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAiSE'01)* (2001), K. Dittrich, A. Geppert, and M. Norrie, Eds., vol. 2068 of *lncs*, springer, pp. 157–170.

[35] **Dellinger, R. P., Levy, M. M., Rhodes, A., Annane, D., Gerlach, H., Opal, S. M., Sevransky, J. E., Sprung, C. L., Douglas, I. S., Jaeschke, R., Osborn, T. M., Nunnally, M. E., Townsend, S. R., Reinhart, K., Kleinpell, R. M., Angus, D. C., Deutschman, C. S., Machado, F. R., Rubenfeld, G. D., Webb, S., Beale, R. J., Vincent, J.-L., Moreno, R., and Surviving Sepsis Campaign Guidelines Committee including The Pediatric Subgroup**. Surviving Sepsis Campaign: International Guidelines for Management of Severe Sepsis and Septic Shock, 2012. *Intensive Care Medicine 39*, 2 (feb 2013), 165–228.

[36] **Desel, J.** *Validation of Process Models by Construction of Process Nets*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000, pp. 110–128.

[37] **Desel, J., and Esparza, J.** *Free Choice Petri Nets*, vol. 40 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, UK, 1995.

[38] **Desel, J., Juhás, G., Lorenz, R., and Neumair, C.** Modelling and validation with viptool. In *International Conference on Business Process Management* (2003), Springer Berlin Heidelberg, pp. 380–389.

[39] **Dixit, P. M., Garcia Caballero, H. S., A., C., Hompes, B. F. A., Buijs, J. C. A. M., and van der Aalst, W.** Enabling interactive process analysis with process mining and visual analytics. In *Proceedings of the 10th International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 5: ACP, (BIOSTEC 2017)* (2017), INSTICC, SciTePress, pp. 573–584.

[40] **Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T.** Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning* (2014), pp. 647–655.

[41] **Ellis, C.** Information control nets: A mathematical model of office information flow. In *Proceedings of the Conference on Simulation, Measurement and Modeling of Computer Systems* (Boulder, Colorado, USA, 1979), ACM Press, pp. 225–240.

[42] **Enlitic**. Bridging human and artificial intelligence to advance medical diagnostics. `https://www.enlitic.com/`. Accessed on April 2021.

[43] **Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al.** A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd* (1996), vol. 96, pp. 226–231.

[44] **Fang, H., Gupta, S., Iandola, F., Srivastava, R. K., Deng, L., Dollár, P., Gao, J., He, X., Mitchell, M., and Platt, J. C.** From captions to visual concepts and back. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 1473–1482.

[45] **Fong, R. C., and Vedaldi, A.** Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 3429–3437.

[46] **Fonseca, P., van Gilst, M. M., Radha, M., Ross, M., Moreau, A., Cerny, A., Anderer, P., Long, X., van Dijk, J. P., and Overeem, S.** Automatic sleep staging using heart rate variability, body movements, and recurrent neural networks in a sleep disordered population. *Sleep* (04 2020). zsaa048.

[47] **Fozoonmayeh, D., Le, H. V., Wittfoth, E., Geng, C., Ha, N., Wang, J., Vasilenko, M., Ahn, Y., and Woodbridge, D. M.-k.** A scalable smartwatch-based medication intake detection system using distributed machine learning. *Journal of Medical Systems 44*, 4 (2020), 1–14.

[48] **Francis, M., Rich, T., Williamson, T., and Peterson, D.** Effect of an emergency department sepsis protocol on time to antibiotics in severe sepsis. *CJEM 12*, 4 (jul 2010), 303–10.

[49] **Garcia Caballero, H. S., Corvo, A., van Meulen, F., Fonseca, P., Overeem, S., van Wijk, J. J., and Westenberg, M. A.** Persleep: A visual analytics approach for performance assessment of sleep staging models. In *Eurographics Workshop on Visual Computing for Biology and Medicine, VCBM 2021* (2021), The Eurographics Association.

[50] **Garcia Caballero, H. S., Corvò, A., Dixit, P. M., and Westenberg, M. A.** Visual analytics for evaluating clinical pathways. In *2017 IEEE Workshop on Visual Analytics in Healthcare (VAHC)* (2017), pp. 39–46.

[51] **Garcia Caballero, H. S., Westenberg, M. A., and Gebre, B.** Explainability for one dimensional temporal inputs of deep learning models. *Demo at the 1st Workshop on Visualization for AI explainability (VISxAI)* (2018). Online publication.

[52] **Garcia Caballero, H. S., Westenberg, M. A., Gebre, B., and van Wijk, J. J.** V-awake: A visual analytics approach for correcting sleep predictions from deep learning models. vol. 38, pp. 1–12.

[53] **Garcia Caballero, H. S., Westenberg, M. A., Verbeek, H. M. W., and van der Aalst, W. M. P.** Visual analytics for soundness verification of process models. In *Business Process Management Workshops* (Cham, 2018), E. Teniente and M. Weidlich, Eds., Springer International Publishing, pp. 744–756.

[54] **Gleicher, M., Albers, D., Walker, R., Jusufi, I., Hansen, C. D., and Roberts, J. C.** Visual comparison for information visualization. *Information Visualization 10*, 4 (2011), 289–309.

[55] **Gleicher, M., Barve, A., Yu, X., and Heimerl, F.** Boxer: Interactive Comparison of Classifier Results. *Computer Graphics Forum* (2020).

[56] **Goldberger, A. L., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E.** Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *Circulation 101*, 23 (2000), e215–e220.

[57] **Gostelow, K., Cerf, V. G., Estrin, G., and Volansky, S.** Proper termination of flow-of-control in programs involving concurrent processes. In *Proceedings of the ACM annual conference-Volume 2* (1972), pp. 742–754.

[58] **Graham, M., and Kennedy, J.** Using curves to enhance parallel coordinate visualisations. In *Proceedings on Seventh International Conference on Information Visualization, 2003. IV 2003.* (2003), IEEE, pp. 10–16.

[59] **Günther, C. W., and Rozinat, A.** Disco: Discover your processes. *BPM (Demos) 940* (2012), 40–44.

[60] **Günther, C. W., and Van Der Aalst, W. M.** Fuzzy mining–adaptive process simplification based on multi-perspective metrics. In *International conference on business process management* (2007), Springer, pp. 328–343.

[61] **Haas, A., Rossberg, A., Schuff, D. L., Titzer, B. L., Holman, M., Gohman, D., Wagner, L., Zakai, A., and Bastien, J.** Bringing the web up to speed with webassembly. *SIGPLAN Not. 52*, 6 (June 2017), 185–200.

[62] **Hachul, H., Frange, C., Bezerra, A. G., Hirotsu, C., Pires, G. N., Andersen, M. L., Bittencourt, L., and Tufik, S.** The effect of menopause on objective sleep parameters: data from an epidemiologic study in são paulo, brazil. *Maturitas 80*, 2 (2015), 170–178.

[63] **Hagen, H., Nielson, G., and Post, F. H.**, Eds. *Dagstuhl '97, Scientific Visualization* (USA, 1997), IEEE Computer Society.

[64] **Heer, J., Card, S., and Landay, J.** Prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (2005), ACM, pp. 421–430.

[65] **Hohman, F. M., Kahng, M., Pienta, R., and Chau, D. H.** Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics* (2018).

[66] **Hotelling, H.** Analysis of a complex of statistical variables into principal components. *Journal of educational psychology 24*, 6 (1933), 417.

[67] **Howell, M. D., Davis, A. M., and G, S.** Management of Sepsis and Septic Shock. *JAMA 317*, 8 (feb 2017), 847.

[68] **Hua, K.-L., Hsu, C.-H., Hidayati, S. C., Cheng, W.-H., and Chen, Y.-J.** Computer-aided classification of lung nodules on computed tomography images via deep learning technique. *OncoTargets and therapy 8* (2015).

[69] **Huang, S. H., LePendu, P., Iyer, S. V., Tai-Seale, M., Carrell, D., and Shah, N. H.** Toward personalizing treatment for depression: predicting diagnosis and severity. *Journal of the American Medical Informatics Association 21*, 6 (2014), 1069–1075.

[70] **Ibáñez, V., Silva, J., and Cauli, O.** A survey on sleep assessment methods. *PeerJ 6* (2018), e4849.

[71] **Iliinsky, N., and Steele, J.** *Designing data visualizations: Representing informational Relationships*. "O'Reilly Media, Inc.", 2011.

[72] **Imtiaz, S. A., and Rodriguez-Villegas, E.** Recommendations for performance assessment of automatic sleep staging algorithms. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (2014), pp. 5044–5047.

[73] **Jolliffe, I.** Principal component analysis. In *International encyclopedia of statistical science*. Springer, 2011, pp. 1094–1096.

[74] **Kahng, M., Andrews, P. Y., Kalro, A., and Chau, D. H. P.** Activis: Visual exploration of industry-scale deep neural network models. *IEEE transactions on visualization and computer graphics 24*, 1 (2018), 88–97.

[75] **Karpathy, A., Johnson, J., and Fei-Fei, L.** Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078* (2015).

[76] **Keim, D., Andrienko, G., Fekete, J.-D., Görg, C., Kohlhammer, J., and Melançon, G.** *Visual Analytics: Definition, Process, and Challenges*. Springer Berlin Heidelberg, 2008, pp. 154–175.

[77] **Keim, D., Kohlhammer, J., Ellis, G., and Mansmann, F.** Mastering the information age: solving problems with visual analytics.

[78] **Keim, D. A.** Visual exploration of large data sets. *Communications of the ACM 44*, 8 (2001), 38–44.

[79] **Keim, D. A.** Information visualization and visual data mining. *IEEE transactions on Visualization and Computer Graphics 8*, 1 (2002), 1–8.

[80] **Kemp, B., Värri, A., Rosa, A. C., Nielsen, K. D., and Gade, J.** A simple format for exchange of digitized polygraphic recordings. *Electroencephalography and clinical neurophysiology 82*, 5 (1992), 391–393.

[81] **Kemp, B., Zwinderman, A. H., Tuk, B., Kamphuisen, H. A., and Oberye, J. J.** Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the eeg. *IEEE Transactions on Biomedical Engineering 47*, 9 (2000), 1185–1194.

[82] **Kirchner, K., Herzberg, N., Rogge-Solti, A., and Weske, M.** Embedding conformance checking in a process intelligence system in hospital environments. In *Process support and knowledge representation in health care*. Springer, 2012, pp. 126–139.

[83] **Krizhevsky, A., Sutskever, I., and Hinton, G. E.** Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (2012), pp. 1097–1105.

[84] **Kruskal, J. B.** Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika 29*, 1 (1964), 1–27.

[85] **Kwon, B. C., Anand, V., Severson, K. A., Ghosh, S., Sun, Z., Frohnert, B. I., Lundgren, M., and Ng, K.** Dpvis: Visual analytics with hidden markov models for disease progression pathways. *IEEE Transactions on Visualization and Computer Graphics* (2020), 1–1. Early access.

[86] **Kwon, B. C., Choi, M.-J., Kim, J. T., Choi, E., Kim, Y. B., Kwon, S., Sun, J., and Choo, J.** Retainvis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 299–309.

[87] **Kwon, J.-m., Kim, K.-H., Eisen, H. J., Cho, Y., Jeon, K.-H., Lee, S. Y., Park, J., and Oh, B.-H.** Artificial intelligence assessment for early detection of heart failure with preserved ejection fraction based on electrocardiographic features. *European Heart Journal-Digital Health 2*, 1 (2021), 106–116.

[88] **Lajnef, T., Chaibi, S., Ruby, P., Aguera, P.-E., Eichenlaub, J.-B., Samet, M., Kachouri, A., and Jerbi, K.** Learning machines and sleeping brains: automatic sleep stage classification using decision-tree multi-class support vector machines. *Journal of neuroscience methods 250* (2015), 94–105.

[89] **Lang, M., Bürkle, T., Laumann, S., and Prokosch, H.-U.** Process mining for clinical workflows: challenges and current limitations. In *MIE* (2008), vol. 136, pp. 229–234.

[90] **Längkvist, M., Karlsson, L., and Loutfi, A.** Sleep stage classification using unsupervised feature learning. *Advances in Artificial Neural Systems 2012* (2012), 5.

[91] **Liang, S.-F., Kuo, C.-E., Hu, Y.-H., Pan, Y.-H., and Wang, Y.-H.** Automatic stage scoring of single-channel sleep eeg by using multiscale entropy and autoregressive models. *IEEE Transactions on Instrumentation and Measurement 61*, 6 (2012), 1649–1657.

[92] **Liu, M., Shi, J., Cao, K., Zhu, J., and Liu, S.** Analyzing the training processes of deep generative models. *IEEE transactions on visualization and computer graphics 24*, 1 (2018), 77–87.

[93] **Liu, M., Shi, J., Li, Z., Li, C., Zhu, J., and Liu, S.** Towards better analysis of deep convolutional neural networks. *IEEE transactions on visualization and computer graphics 23*, 1 (2017), 91–100.

[94] **Lumiata**. Lumiata: Ai for healthcare, healthcare analytics. `https://www.lumiata.com/`. Accessed on April 2021.

[95] **Ma, J., Ovalle, A., and Woodbridge, D. M.-k.** Medhere: A smartwatch-based medication adherence monitoring system using machine learning and distributed computing. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (2018), IEEE, pp. 4945–4948.

[96] **Maaten, L. v. d., and Hinton, G.** Visualizing data using t-sne. *Journal of machine learning research 9*, Nov (2008), 2579–2605.

[97] **Mahendran, A., and Vedaldi, A.** Salient deconvolutional networks. In *European Conference on Computer Vision* (2016), Springer, pp. 120–135.

[98] **Mahulea, C., Mahulea, L., García-Soriano, J.-M., and Colom, J.-M.** Petri nets with resources for modeling primary healthcare systems. In *2014 18th International Conference on System Theory, Control and Computing (ICSTCC)* (2014), IEEE, pp. 639–644.

[99] **Mannhardt, F.** Sepsis Cases - Event Log, jan 2016.

[100] **Mannhardt, F., and Blinde, D.** Analyzing the trajectories of patients with sepsis using process mining. In *RADAR+EMISA 2017* (2017), vol. 1859 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 72–80.

[101] **Mannhardt, F., De Leoni, M., Reijers, H. A., and Van Der Aalst, W. M.** Balanced multi-perspective checking of process conformance. *Computing 98*, 4 (2016), 407–437.

[102] **Mans, R., Reijers, H., van Genuchten, M., and Wismeijer, D.** Mining processes in dentistry. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium* (2012), pp. 379–388.

[103] **Mans, R. S., Schonenberg, M., Song, M., van der Aalst, W. M., and Bakker, P. J.** Application of process mining in healthcare–a case study in a dutch hospital. In *International joint conference on biomedical engineering systems and technologies* (2008), Springer, pp. 425–438.

[104] **Méndez, M., and Radtke, R. A.** Interactions between sleep and epilepsy. *Journal of clinical neurophysiology 18*, 2 (2001), 106–127.

[105] **Mendling, J., Verbeek, H., van Dongen, B., van der Aalst, W., and Neumann, G.** Detection and prediction of errors in epcs of the sap reference model. *Data & Knowledge Engineering 64*, 1 (2008), 312 – 329. Fourth International Conference on Business Process Management (BPM 2006)8th International Conference on Enterprise Information Systems (ICEIS' 2006)Four selected and extended papersThree selected and extended papers.

[106] **Michelis, G., Ellis, C., and Memmi, G.**, Eds. *Proceedings of the Second Workshop on Computer-Supported Cooperative Work, Petri nets and Related Formalisms* (Zaragoza, Spain, June 1994).

[107] **Ming, Y., Cao, S., Zhang, R., Li, Z., Chen, Y., Song, Y., and Qu, H.** Understanding hidden memories of recurrent neural networks. In *2017 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2017), IEEE, pp. 13–24.

[108] **Mitchell, T. M.** *Machine Learning*, 1 ed. McGraw-Hill, Inc., USA, 1997.

[109] **Monroe, M., Rongjian Lan, R., Hanseung Lee, H., Plaisant, C., and Shneiderman, B.** Temporal Event Sequence Simplification. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (dec 2013), 2227–2236.

[110] **Monroe, M., Wongsuphasawat, K., Plaisant, C., Shneiderman, B., Millstein, J., and Gold, S.** Exploring point and interval event patterns: Display methods and interactive visual query, 2012.

[111] **Moody, D.** The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering 35*, 6 (2009), 756–779.

[112] **Morrell, M. J., Finn, L., Kim, H., Peppard, P. E., Safwan Badr, M., and Young, T.** Sleep fragmentation, awake blood pressure, and sleep-disordered breathing in a population-based study. *American journal of respiratory and critical care medicine 162*, 6 (2000), 2091–2096.

[113] **Moser, D., Anderer, P., Gruber, G., Parapatics, S., Loretz, E., Boeck, M., Kloesch, G., Heller, E., Schmidt, A., Danker-Hopfe, H., et al.** Sleep classification according to aasm and rechtschaffen & kales: effects on sleep scoring parameters. *Sleep 32*, 2 (2009), 139–149.

[114] **Munzner, T.** *Visualization analysis and design*. CRC press, 2014.

[115] **Murata, T.** Petri nets: Properties, analysis and applications. *Proceedings of the IEEE 77*, 4 (Apr. 1989), 541–580.

[116] Guidelines for the management of sepsis (including neutropenic sepsis). [Online; accessed 15-May-2017].

[117] **Nichols, J. A., Chan, H. W. H., and Baker, M. A.** Machine learning: applications of artificial intelligence to imaging and diagnosis. *Biophysical reviews 11*, 1 (2019), 111–118.

[118] **Ning, F., Delhomme, D., LeCun, Y., Piano, F., Bottou, L., and Barbano, P. E.** Toward automatic phenotyping of developing embryos from videos. *IEEE Transactions on Image Processing 14*, 9 (2005), 1360–1371.

[119] **of Sleep Medicine, A. A., et al.** The aasm manual for the scoring of sleep and associated events: rules, terminology and technical specifications. *Westchester, IL: American Academy of Sleep Medicine 23* (2007).

[120] **Orr, W. C.** Utilization of polysomnography in the assessment of sleep disorders. *The Medical Clinics of North America 69*, 6 (1985), 1153–1167.

[121] **Ouyang, C., Verbeek, H., van der Aalst, W. M., Breutel, S., Dumas, M., and ter Hofstede, A.** Formal semantics and analysis of control flow in ws-bpel. *Science of Computer Programming 67*, 2 (2007), 162 – 198.

[122] **Patocka, C., Turner, J., Xue, X., and Segal, E.** Evaluation of an emergency department triage screening tool for suspected severe sepsis and septic shock. *Journal for Healthcare Quality 36*, 1 (2014), 52–61.

[123] **Pearson, K.** Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2*, 11 (1901), 559–572.

[124] **Peterson, J.** *Petri Net Theory and the Modeling of Systems*. Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1981.

[125] **Petri, C.** *Kommunikation mit Automaten*. PhD thesis, Institut für instrumentelle Mathematik, Bonn, Germany, 1962. In German.

[126] **Pezzotti, N., Höllt, T., Van Gemert, J., Lelieveldt, B. P., Eisemann, E., and Vilanova, A.** Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE transactions on visualization and computer graphics 24*, 1 (2018), 98–108.

[127] **Phan, H., Andreotti, F., Cooray, N., Chén, O. Y., and De Vos, M.** Seqsleepnet: end-to-end hierarchical recurrent neural network for sequence-to-sequence automatic sleep staging. *IEEE Transactions on Neural Systems and Rehabilitation Engineering 27*, 3 (2019), 400–410.

[128] **Pieczkiewicz, D. S., Finkelstein, S. M., and Hertz, M. I.** Design and evaluation of a web-based interactive visualization system for lung transplant home monitoring data. In *AMIA Annual Symposium Proceedings* (oct 2007), vol. 2007, American Medical Informatics Association, pp. 598–602.

[129] **Piéron, H.** *Le problème physiologique du sommeil.* Masson, 1912.

[130] **Polyvyanyy, A., and Weidlich, M.** Towards a compendium of process technologies: The jbpt library for process model analysis. In *Proceedings of the CAiSE'13 Forum at the 25th International Conference on Advanced Information Systems Engineering (CAiSE)* (2013), Sun SITE Central Europe, pp. 106–113.

[131] **Raidou, R. G., Marcelis, F. J., Breeuwer, M., Gröller, M. E., Vilanova, A., and van de Wetering, H. M.** Visual analytics for the exploration and assessment of segmentation errors. *VCBM 16* (2016), 7–9.

[132] **Ratner, M.** IBM's watson group signs up genomics partners. *Nature Biotechnology 33*, 1, 10–11.

[133] **Rauber, P. E., Fadel, S. G., Falcao, A. X., and Telea, A. C.** Visualizing the hidden activity of artificial neural networks. *IEEE transactions on visualization and computer graphics 23*, 1 (2017), 101–110.

[134] **Rechtschaffen, A., and Kales, A.** A manual of standardized techniques and scoring systems for sleep stages of human subjects. *Brain information Service and Brain Research Institute, Los Angeles, CA, USA* (1968).

[135] **Reisig, W.** *Petri Nets: An Introduction*, vol. 4 of *EATCS Monographs on Theoretical Computer Science*. springer, 1985.

[136] **Ren, D., Amershi, S., Lee, B., Suh, J., and Williams, J. D.** Squares: Supporting interactive performance analysis for multiclass classifiers. *IEEE transactions on visualization and computer graphics 23*, 1 (2017), 61–70.

[137] **Rojas, E., Munoz-Gama, J., Sepúlveda, M., and Capurro, D.** Process mining in healthcare: A literature review. *Journal of biomedical informatics 61* (2016), 224–236.

[138] **Rumelhart, D. E., Hinton, G. E., and Williams, R. J.** Learning representations by back-propagating errors. *nature 323*, 6088 (1986), 533–536.

[139] **Sadiq, W., and Orlowska, M.** Analyzing process models using graph reduction techniques. *Information Systems 25*, 2 (2000), 117–134.

[140] **Sadiq, W., Sadiq, W., Orlowska, M. E., and Orlowska, M. E.** Modeling and verification of workflow graphs. Tech. rep., 1996.

[141] **Sainath, T. N., Vinyals, O., Senior, A., and Sak, H.** Convolutional, long short-term memory, fully connected deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP)* (2015), IEEE, pp. 4580–4584.

[142] **Scheepens, R., Michels, S., van de Wetering, H., and van Wijk, J. J.** Rationale visualization for safety and security. *Computer Graphics Forum 34*, 3 (2015), 191–200.

[143] **Schmidhuber, J.** Deep learning in neural networks: An overview. *Neural networks 61* (2015), 85–117.

[144] **Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D.** Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV* (2017), pp. 618–626.

[145] **Shih, A., and Schoenbaum, S. C.** Measuring Hospital Performance: The Importance of Process Measures. *The Commonwealth Fund* (July 2007).

[146] **Shneiderman, B.** The eyes have it: A task by data type taxonomy for information visualizations. In *The Craft of Information Visualization*, B. B. BEDERSON and B. SHNEIDERMAN, Eds., Interactive Technologies. Morgan Kaufmann, San Francisco, 2003, pp. 364–371.

[147] **Siddiqui, M. K., Morales-Menendez, R., Huang, X., and Hussain, N.** A review of epileptic seizure detection using machine learning classifiers. *Brain Informatics 7*, 1 (2020), 1–18.

[148] **Simonyan, K., Vedaldi, A., and Zisserman, A.** Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).

[149] **Simonyan, K., and Zisserman, A.** Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[150] **Sirinukunwattana, K., Raza, S. E. A., Tsang, Y.-W., Snead, D. R., Cree, I. A., and Rajpoot, N. M.** Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images. *IEEE transactions on medical imaging 35*, 5 (2016), 1196–1206.

[151] **Smith, J. R., Negin, M., and Nevis, A. H.** Automatic analysis of sleep electroencephalograms by hybrid computation. *IEEE transactions on systems science and cybernetics 5*, 4 (1969), 278–284.

[152] **Smith, T. F., Waterman, M. S., et al.** Identification of common molecular subsequences. *Journal of molecular biology 147*, 1 (1981), 195–197.

[153] **Somers, V. K., White, D. P., Amin, R., Abraham, W. T., Costa, F., Culebras, A., Daniels, S., Floras, J. S., Hunt, C. E., Olson, L. J., et al.** Sleep apnea and cardiovascular disease: An american heart association/american college of cardiology foundation scientific statement from the american heart association council for high blood pressure research professional education committee, council on clinical cardiology,

stroke council, and council on cardiovascular nursing in collaboration with the national heart, lung, and blood institute national center on sleep disorders research (national institutes of health). *Journal of the American College of Cardiology 52*, 8 (2008), 686–717.

[154] **Song, M., and van der Aalst, W. M.** Supporting process mining by showing events at a glance. In *Proceedings of the 17th Annual Workshop on Information Technologies and Systems (WITS)* (2007), pp. 139–145.

[155] **Spence, R.** *Information visualization*, vol. 1. Springer, 2001.

[156] **Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M.** Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806* (2014).

[157] **Steinberg, R., Günther, W., Stiltz, I., and Rondot, P.** Eeg-mapping during music stimulation. *Psychomusicology: A Journal of Research in Music Cognition 11*, 2 (1992), 157.

[158] **Stephansen, J. B., Olesen, A. N., Olsen, M., Ambati, A., Leary, E. B., Moore, H. E., Carrillo, O., Lin, L., Han, F., Yan, H., et al.** Neural network analysis of sleep stages enables efficient diagnosis of narcolepsy. *Nature communications 9*, 1 (2018), 1–15.

[159] **Stolper, C. D., Perer, A., and Gotz, D.** Progressive visual analytics: User-driven visual exploration of in-progress analytics. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 1653–1662.

[160] **Strobelt, H., Gehrmann, S., Behrisch, M., Perer, A., Pfister, H., and Rush, A. M.** Seq2seq-vis: A visual debugging tool for sequence-to-sequence models. *IEEE transactions on visualization and computer graphics 25*, 1 (2019), 353–363.

[161] **Strobelt, H., Gehrmann, S., Pfister, H., and Rush, A. M.** Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE transactions on visualization and computer graphics 24*, 1 (2018), 667–676.

[162] **Sugiyama, K., Tagawa, S., and Toda, M.** Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernetics 11*, 2 (1981), 109–125.

[163] **Supratak, A., Dong, H., Wu, C., and Guo, Y.** Deepsleepnet: A model for automatic sleep stage scoring based on raw single-channel eeg. *IEEE Transactions on Neural Systems and Rehabilitation Engineering 25*, 11 (2017), 1998–2008.

[164] **Sutherland, J. V., van den Heuvel, W.-J., Ganous, T., Burton, M. M., and Kumar, A.** Towards an intelligent hospital environment: OR of the future. *Studies in health technology and informatics 118* (2005), 278–312.

[165] **Treisman, A.** Preattentive processing in vision. *Computer vision, graphics, and image processing 31*, 2 (1985), 156–177.

[166] **Trzeciak, S., Dellinger, R. P., Abate, N. L., Cowan, R. M., Stauss, M., Kilgannon, J. H., Zanotti, S., and Parrillo, J. E.** Translating Research to Clinical Practice. *Chest 129*, 2 (feb 2006), 225–232.

[167] **Tsinalis, O., Matthews, P. M., Guo, Y., and Zafeiriou, S.** Automatic sleep stage scoring with single-channel eeg using convolutional neural networks. *arXiv preprint arXiv:1610.01683* (2016).

[168] **UiPath**. Uipath process mining. `https://www.uipath.com/product/process-mining`. Accessed on April 2021.

[169] **Urban, G., Tripathi, P., Alkayali, T., Mittal, M., Jalali, F., Karnes, W., and Baldi, P.** Deep learning localizes and identifies polyps in real time with 96% accuracy in screening colonoscopy. *Gastroenterology* (2018).

[170] **Van Der Aalst, W.** Process mining. *Communications of the ACM 55*, 8 (2012), 76–83.

[171] **van der Aalst, W.** Data Science in Action. In *Process Mining*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016, pp. 3–23.

[172] **van der Aalst, W., Adriansyah, A., and van Dongen, B.** Replaying history on process models for conformance checking and performance analysis. *Wiley Int. Rev. Data Min. and Knowl. Disc. 2*, 2 (Mar. 2012), 182–192.

[173] **van der Aalst, W. M.** Business process management as the "killer app" for petri nets. *Software & Systems Modeling 14*, 2 (2015), 685–691.

[174] **Van der Aalst, W. M., and Weijters, A. J.** Process mining: a research agenda, 2004.

[175] **van der Aalst, W. M. P.** *Process Mining: Data science in Action*. Springer-Verlag, Berlin, Germany, 2016.

[176] **van der Westhuizen, J., and Lasenby, J.** Techniques for visualizing LSTMs applied to electrocardiograms. *ArXiv e-prints* (May 2017).

[177] **van Dongen, B., de Medeiros, A., Verbeek, H., Weijters, A., and van der Aalst, W.** The prom framework: A new era in process mining tool support. In *Applications and Theory of Petri Nets 2005*. Springer, Berlin, Heidelberg, 2005, pp. 444–454.

[178] **van Dongen, B., Jansen-Vullers, M., Verbeek, H., and van der Aalst, W.** Verification of the sap reference models using epc reduction, state-space analysis, and invariants. *Comput. Ind. 58*, 6 (Aug. 2007), 578–601.

[179] **van Dongen, B., van der Aalst, W., and Verbeek, H.** *Verification of EPCs: Using Reduction Rules and Petri Nets*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 372–386.

[180] **van Hee, K., Sidorova, N., and Voorhoeve, M.** *Generalised Soundness of Workflow Nets Is Decidable*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 197–215.

[181] **Verbeek, H.** *Verification of WF-nets*. Eindhoven University of Technology Eindhoven, the Netherlands, 2004.

[182] **Verbeek, H., Basten, T., and Aalst, W.** Diagnosing workflow processes using Woflan. *The Computer Journal 44*, 4 (2001), 246–279.

[183] **Verbeek, H., and Wynn, M.** Verification. In *Modern Business Process Automation: YAWL and its Support Environment*, A. t. Hofstede, W. v. d. Aalst, M. Adams, and N. Russell, Eds., Database Management & Info Retrieval. Springer, Berlin, Germany, 2010, ch. 20, pp. 517–545.

[184] **Vilamala, A., Madsen, K. H., and Hansen, L. K.** Deep convolutional neural networks for interpretable analysis of eeg sleep stage scoring. In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)* (2017), IEEE, pp. 1–6.

[185] **Vinyals, O., Toshev, A., Bengio, S., and Erhan, D.** Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR)* (2015), IEEE, pp. 3156–3164.

[186] **Walter, D. O., Rhodes, J., and Adey, W. R.** Discriminating among states of consciousness by eeg measurements. a study of four subjects. *Electroencephalography and clinical neurophysiology 22*, 1 (1967), 22–29.

[187] **Ward, M.** Multivariate data glyphs: Principles and practice. In *Handbook of data visualization*. Springer, 2008, pp. 179–198.

[188] **Weiskopf, D., Ma, K.-L., van Wijk, J. J., Kosara, R., and Hauser, H.** Scivis, infovis-bridging the community divide. In *Proceedings of the IEEE Visualization Conference. IEEE* (2006), Citeseer.

[189] **Widanagamaachchi, W., Livnat, Y., Bremer, P.-T., and Pascucci, V.** Interactive Visualization and Exploration of Patient Progression in a Hospital Setting. *Proceedings 2016 IEEE VIS Workshop on Visual Analytics in Healthcare, Chicago, IL* (2016).

[190] **Wongsuphasawat, K., and Gotz, D. H.** Outflow: Visualizing patient flow by symptoms and outcome. *Proceedings IEEE VIS 2011 Workshop on Visual Analytics in Healthcare* (2011).

[191] **Wongsuphasawat, K., Guerra Gómez, J. A., Plaisant, C., Wang, T. D., Shneiderman, B., and Taieb-Maimon, M.** LifeFlow: Visualizing an Overview of Event Sequences. *Proceeding CHI '11 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems Pages 1747-1756* (2010).

[192] **Wongsuphasawat, K., Smilkov, D., Wexler, J., Wilson, J., Mané, D., Fritz, D., Krishnan, D., Viégas, F. B., and Wattenberg, M.** Visualizing dataflow graphs of deep learning models in tensorflow. *IEEE transactions on visualization and computer graphics 24*, 1 (2018), 1–12.

[193] **Wulff, K., Gatti, S., Wettstein, J. G., and Foster, R. G.** Sleep and circadian rhythm disruption in psychiatric and neurodegenerative disease. *Nature Reviews Neuroscience 11*, 8 (2010), 589.

[194] **Wynn, M., Verbeek, H., van der Aalst, W. M., ter Hofstede, A., and Edmond, D.** Business process verification - finally a reality! *Business Proc. Manag. Journal 15*, 1 (2009), 74–92.

[195] **Yi, J. S., ah Kang, Y., Stasko, J., and Jacko, J. A.** Toward a deeper understanding of the role of interaction in information visualization. *IEEE transactions on visualization and computer graphics 13*, 6 (2007), 1224–1231.

[196] **Zeiler, M. D., and Fergus, R.** Visualizing and understanding convolutional networks. In *European conference on computer vision* (2014), Springer, pp. 818–833.

[197] **Zhang, J., Bargal, S. A., Lin, Z., Brandt, J., Shen, X., and Sclaroff, S.** Top-down neural attention by excitation backprop. *International Journal of Computer Vision 126*, 10 (2018), 1084–1102.

[198] **Zhang, J., Wang, Y., Molino, P., Li, L., and Ebert, D. S.** Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. *IEEE transactions on visualization and computer graphics 25*, 1 (2018), 364–373.

[199] **Zhang, J., Wu, Y., Bai, J., and Chen, F.** Automatic sleep stage classification based on sparse deep belief net and combination of multiple classifiers. *Transactions of the Institute of Measurement and Control 38*, 4 (2016), 435–451.

[200] **Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A.** Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 2921–2929.

[201] **Zhou, J., Weijters, A., Jansen-Vullers, M., and Mans, R.** *Process mining: acquiring objective process information for healthcare process management with the CRISP-DM framework.* Technische Universiteit Eindhoven, 2009.

[202] **Zordan, M., Costa, R., Macino, G., Fukuhara, C., and Tosini, G.** Circadian clocks: what makes them tick? *Chronobiology international 17*, 4 (2000), 433–451.

Summary

Visual Analytics for Medical Workflow Optimization

Workflows concern the sequences of steps taken to perform a task. Optimizing such workflows is crucial to minimize the time spent, especially in the medical domain. Recent advancements in process mining and machine learning have brought new ways to analyze and improve the medical workflows that physicians use at work. Process mining provides a way to analyze the properties of a workflow, enabling a better understanding. Machine learning leverages the capability of computers to perform human tasks (e.g., classification) to decrease the time spent by doctors in such tasks, enabling them to focus on more prominent tasks (e.g., diagnosis).

In this dissertation, novel visual analytics approaches are presented to understand and optimize medical workflows. In particular, we focus on the following research question:

How can we use interactive visualization and automated techniques to understand and optimize medical workflows?

To answer this question, we show how visualization and automated techniques can be combined to understand medical workflows and to improve specific tasks from both process-centric and machine learning perspectives. For each, we introduce novel visual approaches that enable hypothesis-driven exploration of the output produced by such automated techniques.

The first part of this dissertation introduces approaches that are process-centric, i.e., they combine process mining and visual analytics to improve medical workflows. In chapter 3 we propose a visual approach to analyze workflows in the form of Petri nets. We combine process mining techniques with visualization to enable exploration of properties of Petri nets that are desirable when designing workflows, enabling experts to detect erroneous scenarios and trace these in Petri nets. Chapter 4 presents a visual analytics approach that combines the output of process mining algorithms for sepsis analysis. Specifically, alignments are computed and visualized to understand if the daily work conforms to or deviates from the guidelines proposed to treat sepsis. To this end, event logs are aligned with a given process model representing a guideline, enabling an exploratory process of the deviations from the guideline.

In the second part, we focus on the optimization of specific tasks within workflows via machine learning approaches. Chapter 5 introduces the first visual analytics approach to find misclassifications in ground-truth free environments for sleep staging. In particular, we enable hypothesis generation and verification to find misclassifications and to eventually correct them by means of interaction. We discuss how this approach can be applied in other do-

mains. Chapter 6 presents a case study of explanability for one dimensional temporal inputs in deep learning. Several perturbations are studied to analyze their effect in the generation of saliency maps. These are used to visualize the fragments of the input that are relevant for a model when making a decision. We focus on the sleep staging domain again, but our results can be applied to any temporal input data. In chapter 7 a visual analytics approach is proposed to evaluate the performance of sleep staging classifiers. Understanding the performance for different cohort of patients is critical to utilize such classifiers conveniently. Moreover, the sequential behavior of sleep introduces challenges that are not present in traditional classification problems and need to be addressed. Finally, in chapter 8 we present conclusions and opportunities for future work.

Curriculum Vitæ

Humberto Simón García Caballero was born on the 28th of October 1990 in Cieza, Murcia, Spain. After finishing his secondary education in 2008 at Instituto Diego Tortosa in Cieza, he started the Computer Science program at the University of Murcia. In 2014 he defended his thesis entitled "Acquisition, storage and visualization of ECG signals in Android devices via bluetooth". This work introduced him in the domain of medical data and visualization. Shortly after, he began a Master degree in New Technologies in Computer Science at the same university. He chose the track on "Intelligent and Knowledge Technologies with Aapplications in Medicine", which focused on Artificial Intelligence and Machine Learning applied in the medical domain. In 2015, he started his Master project with the tutoring of the Assistant Professor Manuel Campos Martinez and Jose Manuel Juarez Herrero from University of Murcia, and Dr. Francisco Palacios from University Hospital of Getafe. During his Master project, he designed and evaluated several methods for visualizing antibiograms to support the empirical treatment. He graduated in June 2015 with the dissertation entitled "Antibiogram Visualization for Empirical Treatment Support". The outcome of his master dissertation resulted in his first scientific publication at the XVI Conference of the Spanish Association for Artificial Intelligence.

He started his PhD in November 2015 within the Data Science Flagship, which was part of the collaboration between Eindhoven University of Technology and Philips Research. He performed his research in the visualization group of the former under the supervision of Prof. Michel A. Westenberg and Prof. Jarke J. van Wijk. His research was focused on Visual Analytics for Medical Workflow Optimization. He worked in collaboration with people from multiple disciplines. The results of his work are presented and discussed in this dissertation.

Acknowledgments

The End. This is what comes to my mind after all the work done in these years. Reaching this point could not be possible without the help and support of many other people who I would like to publicly thank in here.

I guess I should start from the beginning. Everything started in October 2015 when I first met Michel Westenberg and Jack van Wijk during the interview of a PhD position. Clearly I remember the good vibe both gave to me. Michel was spending a few jokes, and the three of us were laughing a lot. Throughout the years, Michel and I had many good moments that I will hold for the rest of my life: papers accepted, *borrels*, diners, meetings with a couple of moments to laugh, etc. I cannot put into words all the things that have I learned from him throughout my PhD, but I can say that certainly a lot. From these, I would like to remark one. In 2020, Michel was diagnosed with a life-threatening disease. Sadly and despite all the efforts, this disease took his life in April 2021. In the time span from 2020 to 2021, we had many online meetings. Michel remained always positive, calm, focused and happy. This attitude is one of his greatest lessons to me. I wish things had turned out differently for him, but there are things in life that escape to our control. For all the good moments, for all the knowledge and for his attitude to life, I thank Michel with all my heart.

One of the things that stroke me when I applied for this PhD position in the first place was recognizing the name of Jack van Wijk. Even though I was just a master student at that time with nearly no knowledge about Visualization, I recognized the name of Jack. I will always be grateful to him for accepting me to be part of his research group. He has been a wonderful mentor, always willing to help, even when other duties were pilling up on his stack. His feedback and ideas have always been very detailed and useful. Even in the difficult moments, he was able to find the right words to keep my PhD floating. For all his help, dedication, support and supervision, I would like to thank Jack van Wijk.

Carrying my PhD at Eindhoven University of Technology (TUe) gave me the chance to meet wonderful people. Among my PhD fellows, I would like to thank Alberto Corvò, Bram Cappers, Paul van der Corput, Dennis Collaris, Roeland Scheepens, Stef van den Elzen, Kasper Dinkla, Mickeal Verschoor, Dennis Dingen, Martijn van Dortmont . I can say without any doubt you are some of the smartest people I have met in my life. I shared my office with Alberto, Bram and Paul at first. Two *Mediterranean* and two Dutch. After Paul and Bram had finished their PhDs, Dennis joined Alberto and I. They all made the days in the office more bearable and enjoyable. The discussions we had were always calm and productive. Your ideas were always inspiring and helpful. Thank you all for your time, patience and dedication. I did not have the chance to share much office time with the rest of the guys, but I certainly enjoyed all the chats we had, and all the *vis games* and *vis dinners* with you. You are wonderful people and I am thankful for that. The last year at TUe surely was different. The Covid-19 pandemic changed everything and, unfortunately, I had to communicate with the

new PhDs in an online format. Faizan Siddiqui, Astrid van den Brandt, Linhao Meng, Vidya Prasad and Sanne van der Linden, I wish you the best luck with your PhDs. It was nice to meet you (mostly virtually) and have some Thursday online drinks. I certainly had fun! Still at TUe, I had the chance to meet wonderful people within the visualization group: Andrei Jalba, Huub van de Wetering, Michael Burch and Anna Vilanova. Thank you all for all the talks, coffees, support and help.

The years in Eindhoven ran fast. In fact, many times I found myself saying *time flies*. I know my perception of time passing by has something to do with the people I shared my daily life. Alberto Corvò and Alok Dixit, I am absolutely sure I would have not reached this point without you. I am happy to say that apart from program fellows, you became my friends. The memories we have shared are endless and so is my gratitude. We have travelled together, we have gone through difficult moments and celebrated achievements together. *Los kiddos*, as well as *Los bertos* made a couple of conferences different for some people. Thank you, *kiddones*! Throughout the years in Eindhoven, I made many friends. Alessia, Alice, Antonio, Beppe, Carmine, Cristian, Daniela, Demetrio, Elaine, Ema, Fernando, Gianluca, Gianmarco, Marta, Nacho, Nrupa, Paolo, Pasquale, Raj, Teresò, Zandra, Zia, Zio. I had a lot of fun with all of you. Barbeques, concerts, birthday parties, xmas parties, Ireland, the Irish Pub, *pollo* at the market, rides with the bike, karaoke sessions, *a lot* of laughs, drinks and *suspicious* cookies. Thanks, thanks and thanks! I love you all!

Federica, sure you deserve a special spot in this section. We have gone through many things these years. We had mesmerizing moments in Greece, Portugal, India, Spain, Italy, the UK, Belgium, Germany and Netherlands. These are moments that will remain for ever with us. No matter how difficult the situation has been, we have always faced it together and made it through! Thank you for all the wonderful moments (and cannoli) that we have had these years! A little piece of Sicily will always be embedded in me because of you!

To all the members of the committee: Bernhard Preim, Jörn Kohlhammer, Boudewijn van Dongen, Joos Roerdink, Sebastiaan Overeem and Anna Vilanova; thank you all for your time, comments and availability!

Finally, I would like to thank all the people that, one way or another, have crossed their paths with mine during these years. I hope you too hold a memory that gets a smile out of your faces when you think of it.

Humberto S. Garcia Caballero
Eindhoven, November 2021

Workflows concern the sequences of steps taken to perform a task. Optimizing such workflows is crucial to minimize the time spent, especially in the medical domain. Recent advancements in process mining and machine learning have brought new ways to analyze and improve the medical workflows that physicians use at work. Process mining provides a way to analyze the properties of a workflow, enabling a better understanding. Machine learning leverages the capability of computers to perform human tasks (e.g., classification) to decrease the time spent by doctors in such tasks, enabling them to focus on more prominent tasks (e.g., diagnosis).

In this dissertation, novel visual analytics approaches are presented to understand and optimize medical workflows. In particular, we focus on the following research question:

How can we use interactive visualization and automated techniques to understand and optimize medical workflows?

To answer this question, we show how visualization and automated techniques can be combined to understand medical workflows and to improve specific tasks from both process-centric and machine learning perspectives. For each, we introduce novel visual approaches that enable hypothesis-driven exploration of the output produced by such automated techniques.