Computers in Biology and Medicine 183 (2024) 109244

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/compbiomed

Enabling machine learning models in alarm fatigue research: Creation of a large relevance-annotated oxygen saturation alarm data set ☆

Jonas Chromik [a,1], Anne Rike Flint [b,1,*], Mona Prendke [b], Bert Arnrich [a], Akira-Sebastian Poncette [b]

[a] Hasso Plattner Institute, Rudolf-Breitscheid-Straße 187, Potsdam, 14482, Brandenburg, Germany
[b] Institute of Medical Informatics at Charité – Universitätsmedizin Berlin, corporate member of Freie Universität Berlin and Humboldt-Universität zu Berlin, Charitéplatz 1, Berlin, 10117, Berlin, Germany

ARTICLE INFO

Dataset link: https://zenodo.org/records/10026961

Keywords:
Algorithmic annotation
Patient monitoring
Alarms
Intensive care
Alarm fatigue

ABSTRACT

Background: Too many unnecessary alarms in the intensive care unit are one of the main reasons for alarm fatigue: Medical staff is overburdened and fails to respond appropriately. This endangers both patients and staff. Currently, there are no algorithms that can determine which alarms are clinically relevant and which are not.
Objective: This paper presents a computer-aided method to automatically determine whether and which interventions followed an alarm. Our algorithm annotates a large data set of oxygen saturation alarms. Previous studies only presented analyses on smaller data sets of manually annotated alarms. Future research can use our large data set of labelled alarms to train machine learning models, for example for alarm prioritisation.
Methods: We propose an alarm annotation algorithm that can efficiently label oxygen saturation alarms from respiratory alarm management by actionability. This algorithm is based on an alarm annotation guideline and works on data from 1961 patients from the hospital information system recorded 06/2019–06/2021. The algorithm analyses a pre-defined time frame after an alarm to determine whether an intervention followed or not. The resulting data set can be used to train machine learning models that predict alarm actionability.
Results: Our open-source algorithm is the first to create a large data set of around 2.5 million relevance-annotated alarms in mere hours. A task that would take years using manual annotation. Our algorithm denotes about 9% of the alarms as actionable. This is in line with previous research. The data set also shows which respiratory management interventions medical staff used to counteract the cause of an alarm.
Conclusion: The data set can be a starting point to reduce the number of unnecessary oxygen saturation alarms. For example, it can serve as a training data set for machine learning models that assess future alarms. The algorithm might be re-used to annotate other alarm data sets as well.

1. Introduction

Too many alarms make the intensive care unit (ICU) a noisy and stressful place [1]. The majority of alarms are non-actionable [2], meaning that medical staff does not react to the alarm by performing an intervention to potentially counteract a physiological deterioration of the patient in a certain time window [3,4].

This must not be confused with alarms being unnecessary, irrelevant, or false, which represent normative statements, responding to the question of whether there *should be* an intervention in response to the alarm, regardless of what actually happened after the alarm.

The ICU is a complex environment where multiple factors affect medical staff and might cause them to respond unexpectedly to alarms, for example, in terms of a delayed response or no response at all. Evaluating alarms in terms of necessity has been common in other high-stakes contexts where disturbance poses a great threat, e.g. driving [5]. For the ICU, these factors include (but are not limited to) high total numbers of alarms, high numbers of technically false or otherwise

☆ This work was partially carried out within the INALO project. INALO is a cooperation project between AICURA medical GmbH, Charité – Universitätsmedizin Berlin, idalab GmbH, and Hasso Plattner Institute. INALO is funded by the German Federal Ministry of Education and Research under grant 16SV8559.

* Corresponding author.
E-mail addresses: jonas.chromik@hpi.de (J. Chromik), anne-rike.flint@charite.de (A.R. Flint), mona.prendke@charite.de (M. Prendke), bert.arnrich@hpi.de (B. Arnrich), akira-sebastian.poncette@charite.de (A.-S. Poncette).

[1] These authors contributed equally.

https://doi.org/10.1016/j.compbiomed.2024.109244
Received 23 January 2024; Received in revised form 1 October 2024; Accepted 1 October 2024
Available online 24 October 2024

unnecessary alarms, and the way alarms are presented to the medical staff [1,2,6,7]. This is part of the *alarm fatigue* problem complex — a major problem in today's intensive care [6]. High numbers of alarms and especially high numbers of unnecessary alarms are two of many potential causes of alarm fatigue, and inappropriate response to alarms by medical staff is one of its many effects [7,8]. Currently, there is no method to efficiently determine whether an alarm is actionable (and hence necessary) or non-actionable (which includes unnecessary alarms). The medical staff does not record whether they reacted to an alarm. This lack of data hindered data-driven alarm fatigue research in the past [9,10].

Objective. We want to find out whether an alarm is actionable. To do this, we retrospectively searched the patient data for interventions that might have been conducted to counteract the cause of the alarms. We focus on oxygen saturation alarms because they are a major contributor to alarm fatigue, as they are both a frequent type of alarm[2] and a type of alarm that is often technically incorrect or clinically irrelevant [11]. However, even limiting our data set to oxygen saturation alarms, this is still a considerable data processing effort that we have to perform automatically using an algorithm. Otherwise, we cannot process the plethora of alarms and interventions present in an ICU. Our work answers the research question "How can we automatically determine whether an oxygen saturation alarm is actionable?" This is important groundwork that is necessary to create a large data set of labelled alarms. Researchers can then use this data set to train machine learning models — for example for alarm prioritisation.

Approach. We use a clinically validated annotation guideline (Section 2) that enables retrospective alarm annotation by clinical relevance as a foundation for this algorithm [3]. In Section 3, we show how we transform the annotation guideline into an executable algorithm to *automatically* annotate large numbers of alarms. In Section 4, we present the resulting data set of annotated alarms and elaborate on the data processing obstacles we encountered. The algorithm annotates a whole year's alarm log with millions of alarms in mere hours. With this data set, we can find out how many alarms might have been actionable and which medical interventions usually happen after an oxygen saturation alarm, as we discuss in Section 5.

2. Materials

We use Python 3.8 as the programming language of choice for the implementation. Python, although comparatively slow [13], is preferentially used for endeavours in data science and engineering as evidenced by the vast range of data science-related modules: We use pandas [14] for data analysis and in-memory data management together with seaborn [15] and Matplotlib [16] to visualise results. Data are stored in a MariaDB 10.6.4 [17] database. SQLAlchemy [18] and PyMySQL [19] connect to and query the database.

2.1. Data set

We use routine data from two surgical-anethesiological intensive care units with 21 beds each in a format that closely resembles the MIMIC-IV clinical data set in terms of tables and columns. Our data are not from the original MIMIC-IV data set but from a retrospective, observational study at the intensive care units in a large university hospital in Germany (ethics approval number: EA1/127/18 Ethikausschuss am Campus Charité Mitte, Chairperson: Prof. Dr. med. R. Morgenstern). The data set was extracted retrospectively from the patient data management systems (PDMS) and transposed to an adapted MIMIC-IV structure [20].

The extracted data contain information on 1961 patients recorded over approximately 24 months from 2019/06/02 to 2021/06/13 with a mean length of ICU stay of 41 days (standard deviation: 39 days, range: 1 to 422 days). Of these patients, 852 are male and 1109 are female. The mean patient age at the start of our research time window in 2019 is 62.84 years (standard deviation 17.15). During one of their hospital stays in the time window 348 patients died with a peak around 75–80 years of age (Fig. A.6).

Table 1 describes the tables we use to annotate the alarms. Our algorithm requires these tables in the structure as specified by MIMIC-IV [21] and the annotation guideline [3] to produce labels. When trying to reproduce our annotation procedure with data from a different hospital, researchers need to provide these tables and potentially even supplement the mapping tables from the annotation guideline with data on the airway devices and ventilation machines that are specific to their hospital.

Table 1
Overview of tables used for annotation. Abbreviations: FiO_2: Fraction of inspired oxygen, PEEP: Positive end-expiratory pressure, P_{insp}: Inspiratory pressure.

Table name	Description
alarm_logs	Contains all patient monitoring alarms with timestamps and cause. This table is not part of the original MIMIC-IV structure.
chartevents	"Contains all charted data for all patients" [21]. This includes all measured vital parameters, airway devices, and ventilator settings. With ventilator settings, we are specifically interested in set numerical ventilator parameters — such as FiO_2, PEEP, P_{insp}. Additionally, we are interested in the mode of the ventilator device. This mode tells us whether the device was active or on standby at that time.
procedureevents	"Contains procedures for patients" [21]. This table informs us about the type of oxygen therapy, ventilation device, and ventilation mode used.

2.2. Annotation guideline

We implement parts of the annotation guideline by Klopfenstein et al. related to oxygen saturation alarms. This annotation guideline elaborately enumerates common interventions that may follow an alarm and thus provides a set of rules identifying if an alarm is clinically actionable (Table 2). It adapts the IEC 60601-1-8:2006/AMD2:2020's definition of "clinically actionable" [4, section 3.44]. An alarm is considered actionable if one of the rules applies within a 30-min post-alarm window. These rules suggest potential interventions that could occur within this time frame, impacting an alarm's actionability. Oxygen saturation alarms, for instance, may trigger changes in oxygen therapy, ventilator settings, airway devices, or respiratory support therapies. To categorise the diverse airway management and respiratory support therapy combinations, the annotation guideline maps them into 10 levels of airway management and 8 levels of respiratory support therapies [3].

3. Methods

We annotate oxygen saturation alarms that occurred in the past and reside in a database. This corresponds to a retrospective, observational study. As outcomes, we are interested in both the fraction of actionable alarms and the interventions that follow an actionable alarm. For performance reasons, we group all relevant data by hospital admission. A hospital admission is a patient's single stay at the hospital. All data

[2] Depending on the specific counting methodology, it is the second most common type of alarm according to Siebig et al. [11] and fourth most common type of alarm according to Graham and Cvach [12].

Table 2
Annotation rules: All rules look for an escalation of a parameter. The parameter might be numerical per se (for example oxygen flow or respiratory rate) or the numerical representation of a more complex constellation — as is the case with airway devices or respiratory support therapies. The alarm is considered actionable when at least one rule applies.
Source: Adapted from [3].

Rule name	Condition	Parameter meaning
Increase of set O_2 flow	$O_{2,pre} < O_{2,post}$	Low-dose oxygen flow
Increase of set FiO_2	$FiO_{2,pre} < FiO_{2,post}$	Fraction of inspired oxygen
Escalate flow	$Flow_{pre} < Flow_{post}$	High-dose oxygen flow
Increase of set RR	$RR_{pre} < RR_{post}$	Respiratory rate
Increase of set PEEP	$PEEP_{pre} < PEEP_{post}$	Positive end-expiratory pressure
Increase of set p_{insp}	$p_{insp,pre} < p_{insp,post}$	Inspiratory pressure
Increase of set p_{supp}	$p_{supp,pre} < p_{supp,post}$	Pressure support
Change of airway device (AD)	AD $Level_{pre}$ < AD $Level_{post}$	Airway management level
Change of respiratory support therapy (RST)	RST $Level_{pre}$ < RST $Level_{post}$	Respiratory support therapy level

Table 3
Minimum and maximum values for numerical parameters. Values are either defined by the manufacturers of the corresponding medical devices or by medical experts.

Parameter	Item ID	Min.	Max.	Unit
O_2	107316	0	16	L min^{-1}
FiO_2	101701	0	100	%
Flow	101705	0	60	L min^{-1}
RR	101702	0	40	min^{-1}
PEEP	101704	0	25	cmH_2O
p_{insp}	101703	0	40	cmH_2O
p_{supp}	101706	0	40	cmH_2O

from one stay are separately gathered, held, and administrated by Python objects called *Admissions*. This concept is important, as a patient can be treated multiple times at the same hospital. We treat different stays separate from each other.

To ensure the soundness of the algorithm and correct implementation, we applied common software engineering practices such as pair programming and internal code review. However, this does not ensure that the data set is correct and complete. During implementation, we encountered data-related obstacles that we describe and address in Section 4.1.

3.1. Data preparation

On creation, the Admission objects gather all relevant data for their corresponding hospital admission from the database. A major obstacle is collecting the relevant data items from different subsystems of the hospital information system while maintaining sufficiently low runtime and runtime complexity. These are data on alarms, set numerical ventilation parameters (for example oxygen flow or fraction of inspired oxygen), airway devices (for example endotracheal tube, oxygen mask), ventilation devices (different models from different manufacturers), and ventilation modes (for example SIMV — Synchronised Intermittent Mandatory Ventilation or CPAP — Continuous Positive Airway Pressure), and device modes (on, off, or standby). To gather alarms, we query the identifier and time for all alarm log entries where an oxygen saturation low alarm started for the hospital admission (hadm) at hand (top left in Fig. 1). The numerical parameters of interest (Table 3) reside in the chartevents table. We filter for nonsensical values – below the minimum value or above the maximum value – while querying the data (top left, below the alarm logs query in Fig. 1).

Airway devices. Gathering data on airway devices has two parts: When a new airway device is introduced, the device's name is written to chartevents. We load the charted time and the airway device's name for each event from the database and add placeholder columns for the ventilation device, the ventilation mode, and the device mode (left in Fig. 1, as "airway device insertions"). We need the placeholder columns later on when we merge airway devices, ventilation devices, ventilation modes, and device modes — to get a holistic image of the respiratory support therapy the patient receives. Additionally, we need to consider when patients have no airway device or when it is removed. The respective information also resides in chartevents and we represent it by setting the airway device's name to an empty string (central in Fig. 1, as "airway device removals").

Ventilation data. The procedures table always records the ventilation device and ventilation mode together. Again, we load the time when the new ventilation device and/or mode was introduced and the corresponding names for the ventilation device and mode. This time, we add placeholder columns for the airway device and the device mode for later merge operations (bottom left in Fig. 1, as "ventilation devices and modes").

The chartevents table holds information on the device mode for the ventilation device. We need the device mode to determine whether the ventilation device is currently active or on standby. This time, we add placeholder columns for airway device, ventilation device, and ventilation mode (central in Fig. 1, as "device on standby?").

Merging airway and ventilation data. Airway devices, airway device removals, procedures, and device modes only make sense when considered together. We merge these data into one table that records all ventilation-related events (bottom central in Fig. 1, specifics in Listing 1). In the SQL queries, we already ensured that the query results have the same columns in the same order. This enables simple concatenation of the query result tables for airway devices, airway device removals, procedures, and device modes. A default event at the beginning of the table helps in cases where the first alarm goes off before the first airway or ventilation device is used on the patient.

To get rid of the NULL values we created in the SQL queries for the dummy columns, we use forward fill to carry over the last recorded value for each column in case there is none in the subsequent row. After this, we have a holistic picture of the patient's respiratory situation with every row.

Device modes and ventilation devices on standby. From device mode, we extract the information on whether the device was on standby. Many different modes can denote standby. For convenience, we cross-reference the device mode against a list of known standby modes to create a boolean value for the standby information. For all airway devices, we add the respective airway device identifier and the airway device level from a mapping table. The identifier is necessary to determine the respiration support therapy level later on. The airway device level tells us how invasive the airway device is — a piece of information required to judge the alarm's relevance. By considering the airway device, ventilation device, and ventilation mode together, we can conclude a respiration support therapy level and a level of invasiveness. If the ventilation device is on standby, we assign a respiration support therapy level of zero — meaning a patient is breathing on their own.

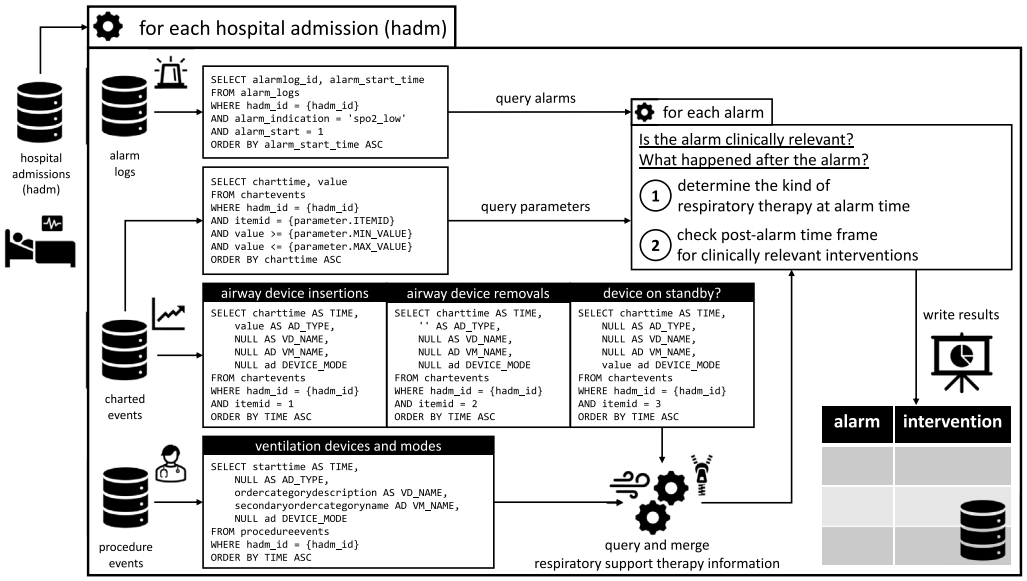

Fig. 1. Schematic overview of the data collation and annotation process. We query different data sources to get a holistic picture of the patient's respiratory state at alarm time. Then, we assess all clinical events in the post-alarm time frame if they are suitable to counteract the state of low oxygen saturation.

Listing 1 Merging data on airway devices, ventilation devices, ventilation modes, and device modes

```
import pandas as pd

def merge_events(
    airway_devices, airway_device_removals, procedures, device_modes):

    events = pd.concat([
            DEFAULT_EVENT,
            airway_devices,
            airway_device_removals,
            procedures,
            device_modes],
        ignore_index=True)

    events = events.sort_values('TIME', ignore_index=True)
    events = events.fillna(method='ffill')

    events['STANDBY'] = events.DEVICE_MODE.apply(
        lambda mode: mode in STANDBY_STRINGS)

    events = events.join(AD_MAPPINGS, on='AD_NAME')

    events['RST_LEVEL'] = events.apply(
        rst_level, axis=1, result_type='reduce')

    events['INVASIVENESS_DEGREE_LEVEL'] = events.apply(
        invasiveness_degree_level, axis=1, result_type='reduce')

    return events
```

3.2. Annotation of oxygen saturation alarms

From a high-level perspective, annotating oxygen saturation alarms is straightforward with the data we already prepared. For each Admission, we iterate over all alarms of this Admission and check whether there are escalations in the airway devices, in the respiration support therapies, or the numerical parameters (top right in Fig. 1, specifics in Listing 2).

Listing 2 High-level view on the annotation procedure

```
def annotate(self):
    for alarmlog_id, _ in self.alarms.iterrows():
        self.annotate_alarm(alarmlog_id)

def annotate_alarm(self, alarmlog_id):
    self.check_airway_device(alarmlog_id)
    self.check_respiration_support_therapies(alarmlog_id)
    self.check_numerical_parameters(alarmlog_id)
```

Airway devices and respiration support therapies. Checking for airway device escalations and respiration support therapy escalations is straightforward with the ventilation events table that we previously established in Listing 1. For both airway devices and respiration support therapies,

we consider the table's last entry before the alarm occurred. We assume that it is identical to the airway device level and the respiration support therapy level at alarm time. Then, we search through the ventilation events in the post-alarm time window of 30 min for the intervention with the highest airway device level or respiration support therapy level, respectively. If the highest airway device level in the post-alarm time window is higher than the airway device level at alarm time, we consider this an airway device escalation; and thus the alarm is actionable. If the highest respiration support therapy level in the post-alarm time window is higher than the respiration support therapy level at alarm time, we consider this a respiration support therapy escalation; and thus the alarm is actionable.

Numerical parameters. Checking for escalations of numerical parameters involves additional checks. For each parameter listed in Table 3, we assess whether it is relevant for the respiration support therapy form that is used at the time of the alarm. For example, the combination of PEEP, p_{insp}, and p_{supp} is only relevant in controlled ventilation settings [22]. Therefore, we ignore these parameters when the patient is breathing spontaneously. If we choose to consider a parameter, we check whether the highest value for the parameter in the post-alarm time window is higher than the parameter's value at alarm time. If so, we consider this an escalation for this parameter; and thus the alarm is actionable.

3.3. Output format

We are ultimately interested in which annotation rules make an alarm actionable. An alarm is actionable, when *at least* one rule applies. When no rule yields a positive result for the alarm, the alarm is not actionable. Initially, the Admission objects store the annotation results. After an Admission object finishes annotating all associated alarms, the annotation results for this admission are written back to the database.

For each rule that made an alarm actionable, we record the alarm's identifier, the rule's name, and the time when the alarm became actionable according to this rule. For rules concerning numerical parameters, we additionally record the parameter's value before the alarm and the value after the alarm. For rules concerning respiration support therapies, we additionally record the airway device, ventilation device, ventilation mode, and device mode, according to the ventilation events table before and after the alarm. This leads to two different tables – one for rules concerning numerical parameters and one for rules concerning respiration support therapies – as different rules call for different sets of information to be recorded. By creating two output tables, we adapt the proposed output format from the annotation guideline [3] to ensure exact bookkeeping. We deem this necessary to thoroughly analyse how medical staff responds to alarms. Just recording whether an alarm is actionable or not would vastly limit the research questions that the resulting data set can answer.

4. Results

We used the algorithm to annotate around 2.5 million oxygen saturation alarms from multiple intensive care units at a large German university hospital. Different intensive care units have vastly different patient and alarm characteristics, as well as alarm policies. To present the annotation results, we visualise annotated alarms from two intensive care units from a period from July 2019 to June 2021.

4.1. Implementation performance & obstacles

A major obstacle is collecting the relevant data items from different tables in the database while maintaining sufficiently low runtime and runtime complexity. To avoid sending many requests to the database management system and querying the same information multiple times, we grouped the alarms by hospital admission. This vastly reduces the runtime of the algorithm from multiple days to annotating the whole alarm data set to mere hours. However, we also ran into data-related obstacles that required us to adapt our implementation accordingly. All of which stem from erroneous or inconsistent documentation in the original electronic health record system. We report the major obstacles and how we addressed them.

Airway device insertions and removals. Concerning airway devices, we noticed that the timing and order of insertions and removal are sometimes nonsensical, for example, airway devices are at times inserted and removed again seconds after. We assume that the medical staff wanted to document the removal of the previous airway device and the insertion of a new one. But since the specific timing is slightly imprecise, the order of the insertions and removal got mixed up as well. We circumvent this issue by rounding down the charted time of airway device removals to full minutes (Listing 3).

Listing 3 Rounding down all airway device removal times (`dt` is a Python datetime object)

```
def floor_to_minutes(dt):
    return dt.replace(second=0, microsecond=0)

airway_device_removals['TIME'] = \
    airway_device_removals.TIME.apply(floor_to_minutes)
```

Inconsistent naming. In the data set we use, medical staff could have entered airway device names manually. This led to many different names and spellings for the same device. To some extent, the annotation guideline already covers different names. But some inconsistencies remain. Especially capitalisation and whitespace characters are problematic. We address this issue by converting all airway device names to lowercase and removing leading and trailing whitespace (Listing 4).

Listing 4 Removing leading and trailing whitespace from airway device names and converting them to lowercase

```
airway_devices['AD_NAME'] = \
    airway_devices['AD_NAME'].str.strip().str.lower()
```

Nonsensical respiration support therapies. The annotation guideline considers only combinations of airway devices and ventilation devices and modes that make sense. It disregards nonsensical combinations, such as controlled ventilation with an oxygen mask. However, according to the documentation in the database, nonsensical combinations can happen. We assume that this is also due to documentation errors. We circumvent this issue by disregarding the airway device and determining respiration support therapy level and invasiveness level only via the ventilation device and ventilation mode.

4.2. Annotation results

The majority of the annotated oxygen saturation alarms for the observed cohort are non-actionable. From 139 870 annotated alarms, only 12 891 are actionable (positive predictive value = 9.22%). Alarm thresholds influence the alarm load in an ICU. Medical personnel set these thresholds and whenever a vital parameter crosses its respective threshold – becomes too high or too low – an alarm goes off. Thresholds might be set according to hypoxemia ranges [23–25]. Our data set includes oxygen saturation alarm thresholds ranging from 70% up to 99%.

Every rule of the annotation guideline led to the classification of an oxygen saturation alarm as actionable at least a few hundred times (Fig. A.7). Most often, only a single rule labelled an alarm as actionable. Rule combinations – where multiple rules apply for one alarm – are possible but occur less frequently. Alarms followed by changes concerning the respiratory support therapy (RST) are usually accompanied by other interventions as well (Fig. 2). Those alarms are

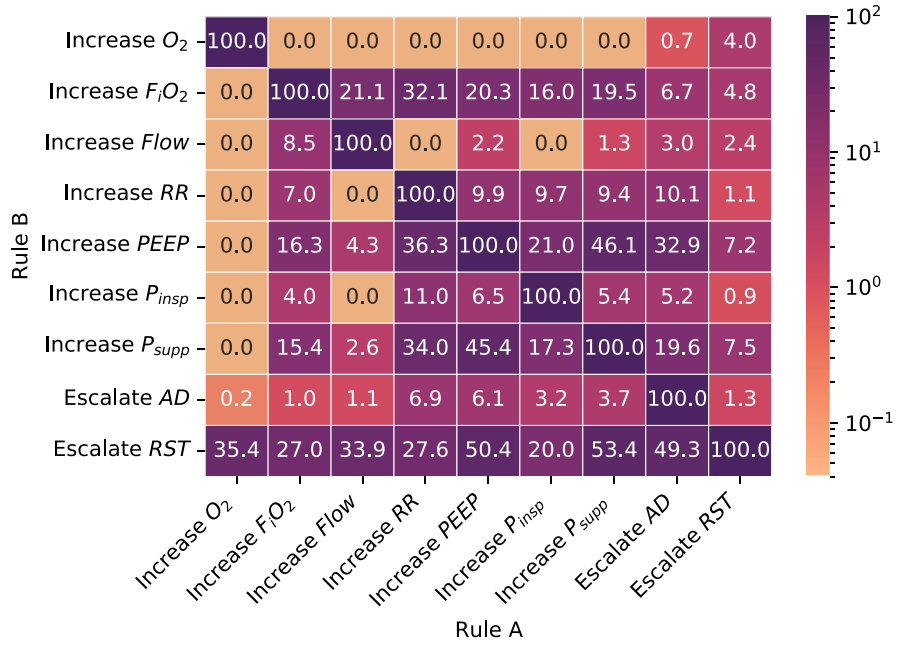

Fig. 2. A heatmap showing the binary co-occurrence of interventions in conditional probability percentages, i.e. the probability that rule B applies when it is already certain that rule A applies for a given alarm ($P(B \mid A)$).

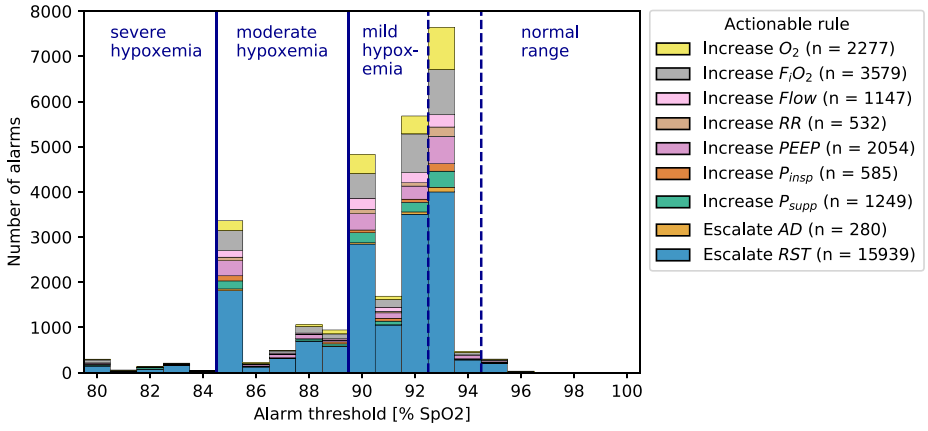

Fig. 3. Absolute distribution of first actionable rule after an oxygen saturation alarm over set oxygen saturation alarm threshold. We can observe that escalating the respiratory support therapy (RST) form is the most common first intervention across alarm thresholds.

likely to be followed by an intervention concerning the airway device (AD), an increase in P_{insp}, or an increase in PEEP. Similarly, an increase in P_{supp} is likely to also involve an increase in PEEP and vice versa. Interventions increasing O_2 rarely involve other interventions.

Up to seven rules classified a single alarm as actionable (Fig. A.8). Most alarms (non-actionable and actionable ones) occur within the set threshold for mild hypoxemia. The vast majority of these alarms are actionable because a single rule (n = 22 036) triggered the annotation algorithm. Regardless of the set threshold, the first intervention in response to an alarm tends to be the escalation of the respiratory support therapy form (n = 15 939, Fig. 3). In comparison, the escalation of the airway device in use only triggered first a few times (n = 280). Exploratory data analysis revealed that the change of an AD is documented infrequently or much later than the observed post-alarm time window of 30 min.

On average, the first intervention that makes an alarm actionable is documented within 15 min after the alarm went off (Fig. 4). The intervention might happen earlier because the medical documentation might entail some delay. Staff most commonly intervenes rather quickly by increasing high-dose oxygen flow (Flow) – if applicable – or by escalating the respiratory support therapy (represented by RST levels, Fig. 5).

5. Discussion

We have shown that we can determine whether an oxygen saturation alarm is actionable or not. Therefore, we can decode respiratory alarm management, specifically which medical interventions follow oxygen saturation alarms. This means an automatic annotation of an alarm data set is possible with the approach described in Section 3. With this, we can automatically create a data set on annotated alarms from an alarm log and corresponding patient data. For now, this implementation focuses on oxygen saturation alarms but it might be extended to other alarms later on. Algorithmically annotating alarms by relevance was done before by Fernandes et al. but their approach was limited to a group of 4 patients only [26]. In our work, the capability to annotate large numbers of alarms is a novel aspect.

The annotation results show that only approximately 9% of the alarms might have been actionable. This number conforms with existing literature: Positive predictive values regarding the relevance of

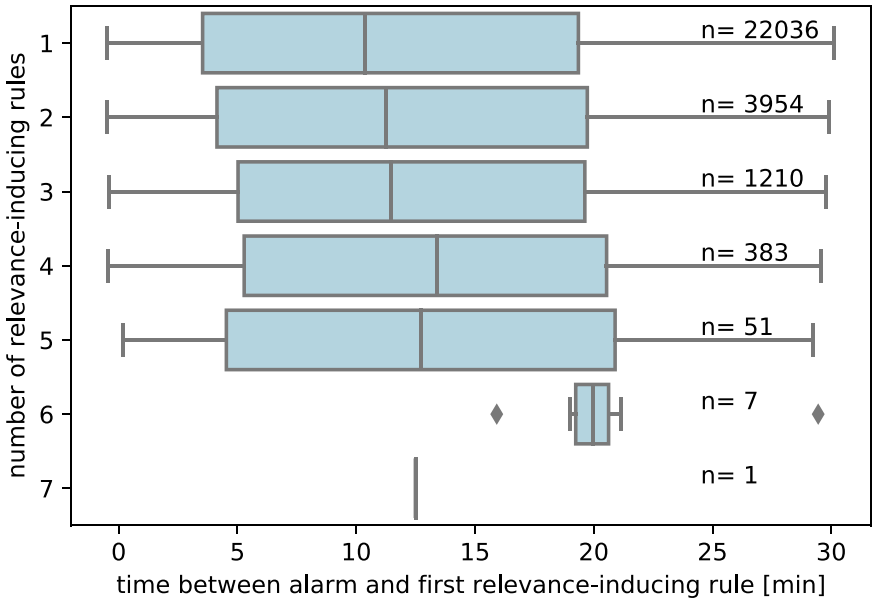

Fig. 4. Distributions of time passed between occurrence of alarm and its first relevance-inducing action subject to the number of relevance-inducing rules. Interestingly, we can observe that the more interventions are taken, the longer it takes for the first intervention to take place.

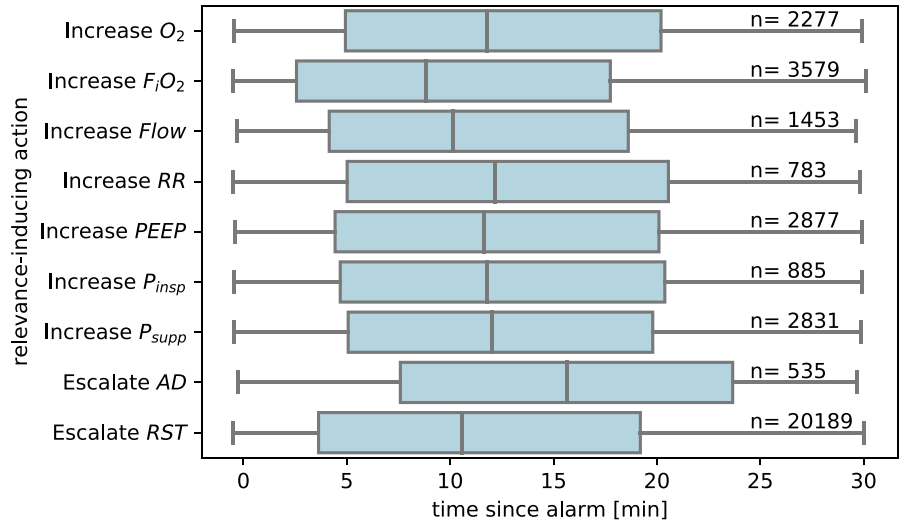

Fig. 5. Distributions of time passed between occurrence of an alarm and its first relevance inducing action by action rule. Please note that this refers to the time an intervention is documented. Prolonged time since alarm could have its cause in tedious documentation.

medical alarms range from <1% [27] to 27% [2,28]. A literature review by Cvach et al. reports positive predictive values between 1% and 20% [6]. A more recent review by Paine et al. reports positive predictive values between 1% and 26% [29]. Thus, our results are well within the expected range.

We approach the problem of alarm fatigue by finding a quantitative proxy for measuring the amount of likely non-actionable alarms. Before our annotation algorithm implementation, large amounts of non-actionable alarms were a mere gut feeling of staff or from studies with only a few thousand alarms. With our results, we show a potential starting point to further investigate influencing factors on high rates of oxygen saturation alarms and their actionability. We have conducted studies on how actionable rates relate to staff workload. We have shown that the number of alarms decreases during the day while the positive predictive value increases [30]. A more surprising result on patient outcomes was that there is no increase in alarm numbers for patients with more severe SOFA (Sequential Organ Failure Assessment) scores [31]. We aim to conduct further research on how actionable and non-actionable rates relate to patient outcomes.

Future research should extend and validate our annotation results with medical expertise, for example, by prospectively validating annotation results. Conversely, our data set can also help to support or question findings from other studies. For example, we know from previous studies, that invasive mechanical ventilation and invasive blood pressure monitoring are likely to lead to higher alarm rates [32]. Future research could compare our retrospectively created data set to findings attained through other methodologies. Eventually, this could lead to a deeper understanding of alarm management in the intensive care unit.

5.1. Limitations

Our approach has two major limitations: One conceptual limitation and one technical (or operational) limitation. Conceptually, our

approach heavily relies on the notion that (1) all interventions in the post-alarm time frame were caused by the alarm and that (2) all alarms that did not cause an intervention did not mandate an intervention. Both assumptions can be challenged. Regarding (1), one might argue that interventions in the post-alarm time frame could have another cause. However, with the information available we cannot perform a test for causality (for example applying the Bradford Hill criteria). Therefore, we have to acquiesce this as an assumption knowing that it might be false. Regarding (2), there might well be alarms that would mandate an intervention but were not responded to with an intervention. Pre-existing alarm fatigue symptoms or high workload might have caused the medical staff to ignore the alarm although intervening would have been the appropriate action. Our approach assumes that the always responds appropriately to every alarm.

As a major technical limitation, documentation errors pose a threat to the validity of our study. Our approach strongly relies on accurately documented interventions and every wrong or missing information skews the results. Low time resolution of the vital parameter measurements is a similar issue, as we cannot analyse how the patient's situation evolved after an alarm. In the future, we might have access to high-resolution vital parameter time series data that provide an even better account of how the patient's situation evolved and how alarms affected the situation.

In the context of advancing data-driven healthcare enabled by artificial intelligence, the significance of data quality and interoperability cannot be overstated. The acquisition of novel electronic healthcare records must diligently consider these crucial aspects. To evaluate and validate our implementation, we compare the annotated alarm data with existing literature on alarm relevance in the intensive care unit. We found the results to be plausible. Additionally, medical experts also checked the results' plausibility — with a positive result as well.

We found all alternative approaches to creating a data set of labelled alarms to be not feasible (compare [3]). Manual labelling through (compensated) crowd-sourcing is not possible, as alarm labelling requires extensive, specialised medical knowledge that only medical experts at the intensive care unit have. Drew et al. demonstrate manual labelling by medical experts, using 4 highly trained professionals to annotate 12,671 arrhythmia alarms [1] which is less than a tenth of our number of annotated alarms. Furthermore, this annotation is not algorithmic and hence not reproducible. Annotating further alarms would – again – require extensive working time of highly trained professionals.

Another common approach is labelling data in real-time as they are recorded. For alarms, this would entail medical experts at the intensive care unit labelling them as they occur. But this is also not feasible, as the medical staff is already overburdened with tasks. Introducing an additional labelling task would skew the results because it entails an extra workload for the medical staff.

Schmidt et al. demonstrated a video-based approach to create a data set of annotated alarms that does not suffer from documentation errors [2]. But if we used video recordings instead of information from the patient data management system we would have needed a human expert that goes through the video recordings and annotates every alarm manually. This would improve accuracy but also limit the data set to much fewer alarms: Our data set consists of 139 870 alarms for just one intensive care unit — over 15 times more compared with the 8975 alarms that Schmidt et al. report [2]. A reduced number of alarms would impair machine learning models trained on the data set [33]. Also, video cameras can have blind spots and occlusions. Schmidt et al. tried to avoid this issue by using two cameras in different positions in the operating theatre. However intensive care units are much larger and more complex environments. Many cameras and endless hours of video analysis would be necessary to cover the whole unit. Which is why we decided against a video-based approach. Instead, we addressed documentation errors by removing seemingly erroneous information, thus potentially increasing the number of non-actionable alarms.

Table 4
Summary table.

What was already known on the topic	• At the intensive care unit, patient monitors notify medical staff through audiovisual alarms when a patient's condition deteriorates. • Most patient monitoring alarms are irrelevant as they do not lead to a medical intervention as a consequence. • But there are no algorithms that can discern which alarms are relevant and which are not. • High numbers of irrelevant alarms lead to alarm fatigue, endangering both patients and staff.
What this study added to our knowledge	• We present an algorithm that can automatically and efficiently label large numbers of oxygen saturation alarms by relevance • The algorithm uses data from hospital information systems and medical guidelines to determine whether an alarm triggered a respiratory management intervention as its consequence or not. • Additionally, the algorithm determines how medical staff reacted to the alarm (if they reacted). • We publish the algorithm and a large data set of labelled oxygen saturation alarms alongside this paper.

5.2. Conclusion & next steps

Alarm fatigue is a major problem in today's intensive care medicine. Patients and staff are at risk of serious health effects through too many, often non-actionable alarms. Algorithmic solutions are necessary to analyse which alarms are actionable and which are not because the sheer number of alarms would overburden every attempt of large-scale manual annotation. We provide such an algorithmic solution.

Conceptually, our algorithm can be used in other hospitals as well as long as this hospital can provide their clinical data in a MIMIC-IV-like format, as proposed by Giesa et al. [20]. In practice, adaptations might be necessary: The airway devices or ventilation devices used at another hospital (or maybe just the names used for these devices) might be different. This would require an update for our airway device level mappings and respiratory support therapy level mappings. These updates need to be done by a medical expert. Apart from these naming issues that constitute an implementation detail that differs between hospitals, we do not see any obstacle preventing the algorithm from being used at other hospitals as well. Further, other clinics might have to adapt the post-alarm time window. We chose a 30-min post-alarm time window after screening the required data points in our EHR. After implementing the annotation algorithm, our results (Figs. 4 and 5) show that the median time between an alarm and the first relevant action is between 10 and 15 min. Future work should include clinicians and technicians from other hospitals to determine the optimal length for the post-alarm time window. This could differ between different hospitals based on implementation details of the respective EHR systems, for example, how often they store certain data.

Next to the algorithm that we provide, researchers can use the vast, annotated alarm data set of oxygen saturation alarms created in this work to help alleviate and counteract alarm fatigue. For example, machine learning models can be trained on these data to prioritise alarms and recommend interventions. Since the data sets differ between hospitals (see paragraph above), the machine learning models will also differ between hospitals and therefore have low external validity. This could pose an implementation barrier. However, since we also publish the annotation algorithm, a separate data set can be created for each hospital to train a specific machine learning model on these data. Another implementation barrier might be the question of what kind of task the machine learning model is supposed to perform. If the model's task is to suppress alarms that are (probably) non-actionable, this would manifest a much higher risk class than just an advisory system and would thus be harder to get authority approval for. However, an advisory system would add to the flood of information that the medical staff

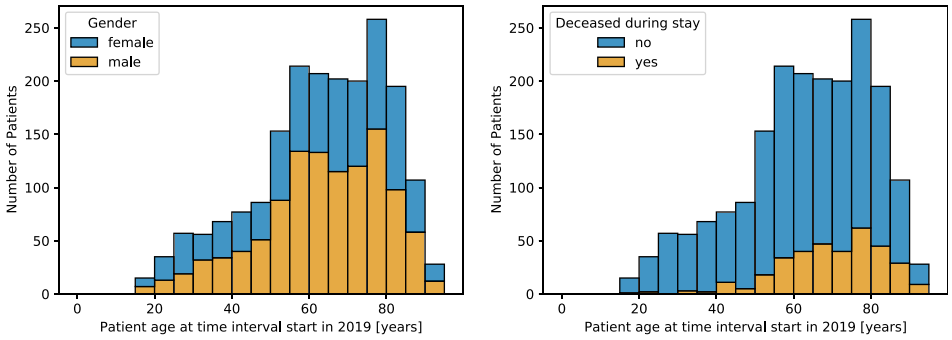

Fig. A.6. A histogram showing the age distribution of our patient population. The histogram is further subdivided by gender (left) and survival (right).

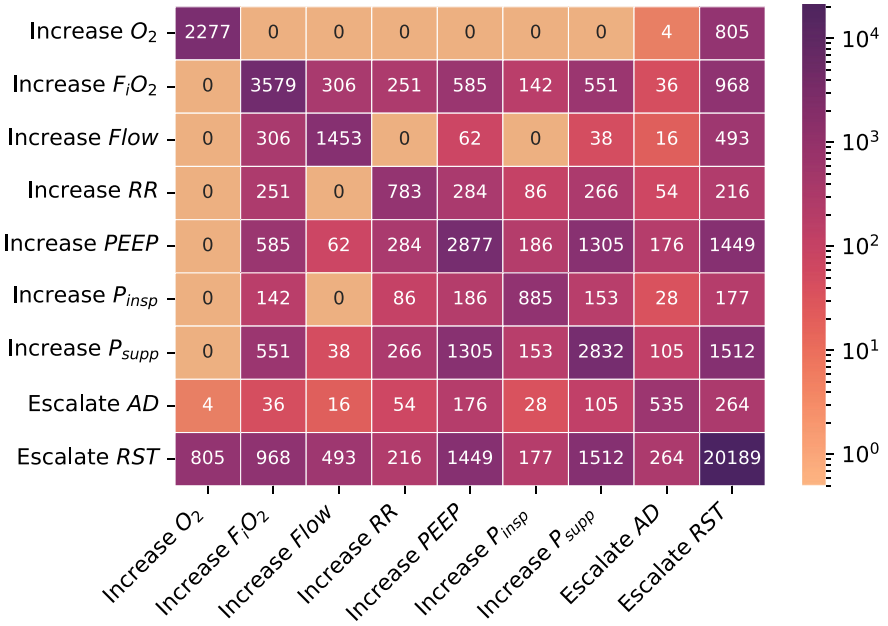

Fig. A.7. A heatmap showing the binary co-occurrence of interventions in total numbers.

has to sort through and thus might be of little practical use. The medical staff that we interviewed told us that they practically always know *how* to respond to an alarm and that they do not need intervention recommendations. The core problem for them is the sheer alarm load. Therefore, an advisory system (as described above) should focus on conveying a prediction on how likely the alarm is actionable, i.e. how clinically relevant the alarm probably is. Since different standardised alarm sounds are already used in medical devices to convey different alarm urgencies [4], we cannot use different alarm sounds to convey the likelihood of actionability. Therefore, we propose a traffic-light-like system where the likelihood of actionability is colour-coded and serves as an additional advisory system that builds upon the existing alarm system.

We provide a large annotated oxygen saturation data set and the algorithm source code here: 10.5281/zenodo.10021845 (see Table 4).

CRediT authorship contribution statement

Jonas Chromik: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation. **Anne Rike Flint:** Writing – review & editing, Visualization, Validation, Software, Project administration, Methodology, Investigation, Data curation. **Mona Prendke:** Writing – review & editing, Visualization. **Bert Arnrich:** Supervision, Resources, Funding acquisition. **Akira-Sebastian Poncette:** Writing – review & editing, Supervision, Resources, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Supplementary material available at https://zenodo.org/records/10026961.

Appendix A. Additional figures

See Figs. A.6–A.8.

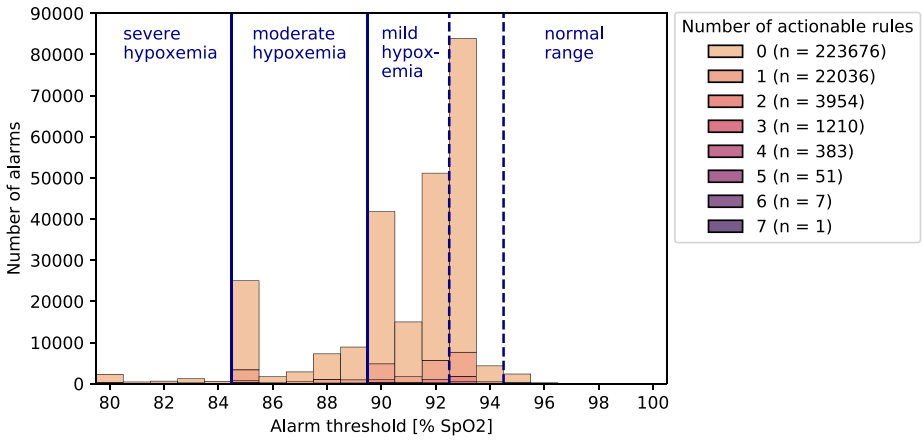

Fig. A.8. Absolute distribution of the number of actionable rules connected to an alarm over set oxygen saturation alarm thresholds. We can observe that the alarm frequency – actionable or not – is subject to the set alarm threshold. Lower limits of common hypoxemia ranges seem to be common alarm thresholds as well and consequently cause many alarms.

References

[1] B.J. Drew, P. Harris, J.K. Zègre-Hemsey, T. Mammone, D. Schindler, R. Salas-Boni, Y. Bai, A. Tinoco, Q. Ding, X. Hu, Insights into the problem of alarm fatigue with physiologic monitor devices: A comprehensive observational study of consecutive intensive care unit patients, PLoS ONE 9 (10) (2014) e110274, http://dx.doi.org/10.1371/journal.pone.0110274.

[2] F. Schmid, M.S. Goepfert, D. Kuhnt, V. Eichhorn, S. Diedrichs, H. Reichenspurner, A.E. Goetz, D.A. Reuter, The wolf is crying in the operating room: Patient monitor and anesthesia workstation alarming patterns during cardiac surgery, Anesth. Analg. 112 (1) (2011) 78–83, http://dx.doi.org/10.1213/ANE.0b013e3181fcc504.

[3] S.A.I. Klopfenstein, A.R. Flint, P. Heeren, M. Prendke, A. Chaoui, T. Ocker, J. Chromik, B. Arnrich, F. Balzer, A.-S. Poncette, How to Annotate Patient Monitoring Alarms in Intensive Care Medicine for Machine Learning, Preprint, 2023, http://dx.doi.org/10.21203/rs.3.rs-2514969/v1, In Review.

[4] International Electrotechnical Commission, IEC 60601-1-8:2006/AMD2:2020, Tech. rep., International Electrotechnical Commission, 2020.

[5] M.N. Lees, J.D. Lee, The influence of distraction and driving context on driver response to imperfect collision warning systems, Ergonomics 50 (2007) http://dx.doi.org/10.1080/00140130701318749.

[6] M. Cvach, Monitor alarm fatigue: An integrative review, Biomed. Instrum. Technol. 46 (4) (2012) 268–277, http://dx.doi.org/10.2345/0899-8205-46.4.268.

[7] M.F. Rayo, S.D. Moffatt-Bruce, Alarm system management: Evidence-based guidance encouraging direct measurement of informativeness to improve alarm response, BMJ Qual. Saf. 24 (4) (2015) 282–286, http://dx.doi.org/10.1136/bmjqs-2014-003373.

[8] M. Wilken, D. Hüske-Kraus, A. Klausen, C. Koch, W. Schlauch, R. Röhrig, Alarm fatigue: Causes and effects, Stud. Health Technol. Inform. 243 (2017) 107–111.

[9] M. Wilken, D. Hüske-Kraus, R. Röhrig, Alarm fatigue: Using alarm data from a patient data monitoring system on an intensive care unit to improve the alarm management, Stud. Health Technol. Inform. 267 (2019) 273–281, http://dx.doi.org/10.3233/SHTI190838.

[10] J. Chromik, S.A.I. Klopfenstein, B. Pfitzner, Z.-C. Sinno, B. Arnrich, F. Balzer, A.-S. Poncette, Computational approaches to alleviate alarm fatigue in intensive care medicine: A systematic literature review, Front. Digit. Health 4 (2022) 843747, http://dx.doi.org/10.3389/fdgth.2022.843747.

[11] S. Siebig, S. Kuhls, M. Imhoff, U. Gather, J. Schölmerich, C.E. Wrede, Intensive care unit alarms—How many do we need? Crit. Care Med. 38 (2) (2010) 451–456, http://dx.doi.org/10.1097/CCM.0b013e3181cb0888.

[12] K.C. Graham, M. Cvach, Monitor alarm fatigue: Standardizing use of physiological monitors and decreasing nuisance alarms, Am. J. Crit. Care: Off. Publ. Am. Assoc. Crit.-Care Nurses 19 (1) (2010) 28–34; quiz 35, http://dx.doi.org/10.4037/ajcc2010651.

[13] The Computer Language Benchmarks Game, Which Programming Language is Fastest? Tech. rep., The Debian Project, Online, 2023.

[14] J. Reback, Jbrockmendel, W. McKinney, J. Van Den Bossche, T. Augspurger, M. Roeschke, S. Hawkins, P. Cloud, Gfyoung, Sinhrks, P. Hoefler, A. Klein, T. Petersen, J. Tratner, C. She, W. Ayd, S. Naveh, J. Darbyshire, M. Garcia, R. Shadrach, J. Schendel, A. Hayden, D. Saxton, M.E. Gorelli, F. Li, M. Zeitlin, V. Jancauskas, A. McMaster, T. Wörtwein, P. Battiston, Pandas-Dev/Pandas: Pandas 1.4.2, Zenodo, 2022, http://dx.doi.org/10.5281/ZENODO.3509134.

[15] M. Waskom, Seaborn: Statistical data visualization, J. Open Source Softw. 6 (60) (2021) 3021, http://dx.doi.org/10.21105/joss.03021.

[16] T.A. Caswell, M. Droettboom, A. Lee, E.S. De Andrade, T. Hoffmann, J. Hunter, J. Klymak, E. Firing, D. Stansby, N. Varoquaux, J.H. Nielsen, B. Root, R. May, P. Elson, J.K. Seppänen, D. Dale, J.-J. Lee, D. McDougall, A. Straw, P. Hobson, Hannah, C. Gohlke, A.F. Vincent, T.S. Yu, E. Ma, S. Silvester, C. Moad, N. Kniazev, E. Ernest, P. Ivanov, Matplotlib/Matplotlib: REL: V3.5.1, Zenodo, 2021, http://dx.doi.org/10.5281/ZENODO.592536.

[17] MariaDB, MariaDB community server 10.6, 2022.

[18] M. Bayer, SQLAlchemy, in: A. Brown, G. Wilson (Eds.), The Architecture of Open Source Applications Volume II: Structure, Scale, and a Few more Fearless Hacks, aosabook.org, Mountain View, 2012, p. 1.

[19] Y. Matsubara, PyMySQL, 2016.

[20] N. Giesa, P. Heeren, S. Klopfenstein, A. Flint, L. Agha-Mir-Salim, A. Poncette, F. Balzer, S. Boie, MIMIC-IV as a clinical data schema, Stud. Health Technol. Inform. 294 (2022) 559–560, http://dx.doi.org/10.3233/SHTI220522.

[21] A. Johnson, L. Bulgarelli, T. Pollard, S. Horng, L.A. Celi, R. Mark, MIMIC-IV, 2020, http://dx.doi.org/10.13026/A3WN-HQ05.

[22] S.A.I. Klopfenstein, A.R. Flint, P. Heeren, M. Prendke, A. Chaoui, T. Ocker, J. Chromik, B. Arnrich, F. Balzer, A.-S. Poncette, Mappings for "how to annotate patient monitoring alarms in intensive care medicine for machine learning", 2023, http://dx.doi.org/10.5281/zenodo.7511032.

[23] M.-A. Blanchet, G. Mercier, A. Delobel, E. Nayet, P.-A. Bouchard, S. Simard, E. L'Her, R.D. Branson, F. Lellouche, Accuracy of multiple pulse oximeters in stable critically ill patients, Respir. Care 68 (5) (2023) 565–574, http://dx.doi.org/10.4187/respcare.10582.

[24] R.N. Smith, R. Hofmeyr, Perioperative comparison of the agreement between a portable fingertip pulse oximeter v. a conventional bedside pulse oximeter in adult patients (COMFORT trial), S. Afr. Med. J. 109 (3) (2019) 154, http://dx.doi.org/10.7196/SAMJ.2019.v109i3.13633.

[25] P.S. Kruger, P.J. Longden, A study of a hospital staff's knowledge of pulse oximetry, Anaesth Intensive Care. 25 (1) (1997) 38–41, http://dx.doi.org/10.1177/0310057X9702500107.

[26] C. Fernandes, S. Miles, C.J.P. Lucena, Detecting false alarms by analyzing alarm-context information: Algorithm development and validation, JMIR Med. Inform. 8 (5) (2020) e15407, http://dx.doi.org/10.2196/15407.

[27] C.L. Tsien, J.C. Fackler, Poor prognosis for existing monitors in the intensive care unit, Crit. Care Med. 25 (4) (1997) 614–619, http://dx.doi.org/10.1097/00003246-199704000-00010.

[28] M.C. Chambrin, P. Ravaux, D. Calvelo-Aros, A. Jaborska, C. Chopin, B. Boniface, Multicentric study of monitoring alarms in the adult intensive care unit (ICU): A descriptive analysis, Intensive Care Med. 25 (12) (1999) 1360–1366, http://dx.doi.org/10.1007/s001340051082.

[29] C.W. Paine, V.V. Goel, E. Ely, C.D. Stave, S. Stemler, M. Zander, C.P. Bonafide, Systematic review of physiologic monitor alarm characteristics and pragmatic interventions to reduce alarm frequency, J. Hosp. Med. 11 (2) (2016) 136–144, http://dx.doi.org/10.1002/jhm.2520.

[30] M. Prendke, A.-R. Flint, K. Rubarth, F. Balzer, A.-S. Poncette, When are alarms most relevant: A temporal analysis of alarm relevance in the intensive care setting, in: ESICM LIVES 2023, in: Intensive Care Medicine Experimental, vol. 11, Springer Nature, 2023, pp. 76–77.

[31] A. Chaoui, A.-R. Flint, K. Rubarth, F. Balzer, A.-S. Poncette, Relationship between SOFA scores and alarm metrics in intensive care units: Implications for alarm fatigue, in: ESICM LIVES 2023, in: Intensive Care Medicine Experimental, vol. 11, Springer Nature, 2023, pp. 257–259.

[32] Z.-C. Sinno, D. Shay, J. Kruppa, S.A. Klopfenstein, N. Giesa, A.R. Flint, P. Herren, F. Scheibe, C. Spies, C. Hinrichs, A. Winter, F. Balzer, A.-S. Poncette, The influence of patient characteristics on the alarm rate in intensive care units: A retrospective cohort study, Sci. Rep. 12 (1) (2022) 21801, http://dx.doi.org/10.1038/s41598-022-26261-4.

[33] A.R. Flint, S.A. Klopfenstein, P. Heeren, F. Balzer, A.-S. Poncette, Utilizing intensive care alarms for machine learning, in: B. Séroussi, P. Weber, F. Dhombres, C. Grouin, J.-D. Liebe, S. Pelayo, A. Pinna, B. Rance, L. Sacchi, A. Ugon, A. Benis, P. Gallos (Eds.), Studies in Health Technology and Informatics, IOS Press, 2022, http://dx.doi.org/10.3233/SHTI220453.